SANDIA REPORT

SAND2015-8857
Unlimited Release
Printed September, 2015

Final Report: Sublinear Algorithms for In-situ and In-transit Data Analysis at Exascale

Janine Bennett, Ali Pinar, C. Seshadhri, David Thompson, Maher Salloum, Ankit Bhagatwala, Jacqueline H. Chen

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online ordering: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2015-8857
Unlimited Release
Printed September, 2015

Final Report: Sublinear Algorithms for In-situ and In-transit Data Analysis at Exascale

Janine Bennett Ali Pinar C. Seshadhri David Thompson
Maher Salloum Ankit Bhagatwala Jacqueline H. Chen

Abstract

Post-Moore's law scaling is creating a disruptive shift in simulation workflows, as saving the entirety of raw data to persistent storage becomes expensive. We are moving away from a post-process centric data analysis paradigm towards a concurrent analysis framework, in which raw simulation data is processed as it is computed. Algorithms must adapt to machines with extreme concurrency, low communication bandwidth, and high memory latency, while operating within the time constraints prescribed by the simulation. Furthermore, input parameters are often data dependent and cannot always be prescribed. The study of sublinear algorithms is a recent development in theoretical computer science and discrete mathematics that has significant potential to provide solutions for these challenges. The approaches of sublinear algorithms address the fundamental mathematical problem of understanding global features of a data set using limited resources. These theoretical ideas align with practical challenges of in-situ and in-transit computation where vast amounts of data must be processed under severe communication and memory constraints. This report details key advancements made in applying sublinear algorithms in-situ to identify features of interest and to enable adaptive workflows over the course of a three year LDRD. Prior to this LDRD, there was no precedent in applying sublinear techniques to large-scale, physics based simulations. This project has definitively demonstrated their efficacy at mitigating high performance computing challenges and highlighted the rich potential for follow-on research opportunities in this space.

Acknowledgment

This work was funded by the Laboratory Directed Research and Development (LDRD) program of Sandia National Laboratories. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Steady improvements in computing resources enable ever more enhanced scientific simulations, however Input/Output (I/O) constraints are impeding their impact. Historically, scientific computing workflows have been defined by three independent stages (see Figure 1.1(a)): 1) a pre-processing stage comprising initialization and set up (for example mesh generation, or initial small-scale test runs); 2) the scientific computation itself (in which data is periodically saved to disk at a prescribed frequency); and 3) post-processing and analysis of data for scientific insights. With improved computing resources scientists are increasing the temporal resolution of their simulations. However, as computational power continues to outpace I/O capabilities, the gap between time steps saved to disk keeps increasing. This compromise in the fidelity of the data being saved to disk makes it impossible to track features with timescales smaller than that of I/O frequency. Moreover, this situation is projected to worsen as we look ahead to future architectures with improvements in computational power continuing to significantly outpace I/O capabilities [48, 2], see Table 1.1.

Consequently, we are seeing a paradigm shift away from the use of prescribed I/O frequencies and post-process-centric data analysis, towards a more flexible concurrent paradigm in which raw simulation data is processed in-situ as it is computed, see Figure 1.1 (b). In spite of the paradigm shift, concurrent processing does not provide a complete solution, as it requires all analysis questions be posed *a priori*. This is not always possible as scientists often analyze their data in an interactive and exploratory fashion. One potential solution, is to store data judiciously for only the time segments that will merit further analysis. However, the problem with this approach is that the computation required to automatically and adaptively make decisions regarding the workflow (e.g,. I/O and/or in-situ data analysis frequencies) based on simulation state can be prohibitively expensive in their own right. Therefore, one of the more pressing fundamental research challenges is the need for efficient, adaptive, data-driven control-flow mechanisms for extreme-scale scientific simulation workflows.

This LDRD research focused on addressing several pressing fundamental high-performance computing (HPC) challenges posed by moving to this new workflow paradigm:

- At what frequency should I/O or analysis be peformed?
- Can we make this decision in an adaptive, data-driven decision at runtime?
- How can we make these decisions quickly and efficiently?
- How do we design efficient analysis algorithms given in-situ constraints?

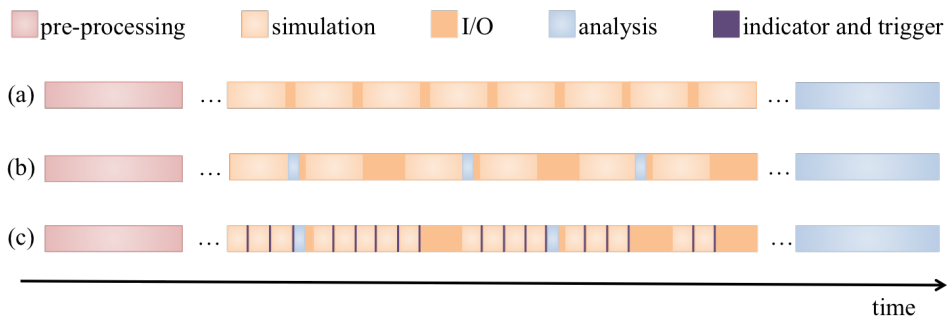


Figure 1.1. (a) An illustration of a traditional workflow made up of 3 stages: 1) pre-procesing, 2) scientific computation and I/O at a prescribed rate, 3) analysis as a post-process. (b) Computational capabilities are outpacing I/O on future architectures, causing a change in workflows as some portion of the analysis moves in-situ. Most current day workflows remain static, with the frequency of I/O and analysis being prescribed upfront by the scientists. (c) This work introduces the use of indicators and triggers to support adaptive workflows. The indicator and trigger are lightweight functions evaluated at a high frequency to make dynamic data-driven control-flow decisions.

Sublinear Algorithms A recent development in theoretical computer science and mathematics is the study of sublinear algorithms, which aim to understand global features of a data set while using limited resources. Often enough, we do not need to look at the entire data to determine some of its important features. The field of sublinear algorithms [20, 40, 42] makes precise the settings when this is possible and combines discrete math and algorithmic techniques with statistical tools to quantify error and give trade-offs with sample sizes. This confidence measure is necessary for adoption of such techniques by the scientific computing community, whose scientific results can be used to make high-impact decisions.

Formally, given a function, $f : D \to R$, we assume for any $x \in D$, we can query $f(x)$. For example, in S3D simulations we have $D = [n]^3$ as a structured grid, and $R = \mathbb{R}$ can be the temperature values. If $D = [n]$ and $R = \{0, 1\}$, then f represents an n-bit binary string. If $R = \{A, T, G, C\}$, f could represent a DNA segment. If $D = [n]^2$, the function could represent a matrix (or a graph). Note D can also be an unstructured grid, modeled as a graph. Similarly, almost any data analysis input can be cast as a collection of functions over a discrete or discretized domain.

We are interested in some *specific* property of f, which is phrased as a yes/no question. For instance, in a jet simulation we can ask if there exists a high-temperature region spanning the x-axis of the grid. How can we determine if f satisfies property $\mathcal{P}$ *without querying all of* f? It is impossible to give an exact answer without knowledge of f. To formalize what can be

Table 1.1. Expected exascale architecture parameters for the design of two "swim lanes" of very different design choices [48, 2]. Note the drastic difference between expected improvements in I/O and compute capacities in both swim lanes.

System Parameter	2011	2018		Factor Change
System Peak	2 Pf/s	1 Ef/s		500
Power	6 MW	$\leq$ 20 MW		3
System Memory	0.3 PB	32-64 PB		100-200
Total Concurrency	225K	1B$\times$ 10	1B $\times$ 100	40000-400000
Node Performance	125 GF	1TF	10 TF	8-80
Node Concurrency	12	1000	10000	83-830
Network Bandwidth	1.5 GB/s	100 GB/s	1000 GB/s	66-660
System Size (nodes)	18700	1000000	100000	50-500
I/O Capacity	15 PB	30-100 PB		20-67
I/O Bandwidth	0.2 TB/s	20-60 TB/s		10-30

inferred by querying $o(|D|)$ values of f, we use a notion of the *distance to* $\mathcal{P}$ [1]. Every function f has a distance to $\mathcal{P}$, denoted ε_f, where $\varepsilon_f = 0$ iff f satisfied $\mathcal{P}$. To provide an exact answer to questions regarding $\mathcal{P}$, we determine whether $\varepsilon_f = 0$ or $\varepsilon_f \neq 0$. However, approximate answers can be given by choosing some error parameter $\varepsilon > 0$ and then determining whether we can distinguish $\varepsilon_f = 0$ from $\varepsilon_f > \varepsilon$. The theory of sublinear algorithms shows whether the latter question can be resolved by an algorithm that samples $o(|D|)$ function values.

For a sublinear algorithm, there are usually three parameters of interest: the number of samples t, the error ε, and the confidence δ. As described earlier, the error is expressed as the distance to $\mathcal{P}$. Analysis shows that for a given t, we can estimate the answer within error ε with a confidence of $> 1 - \delta$. Conversely, given ε, δ, we can compute the number of samples required.

Although at a high level, any question that can be framed in terms of determining global properties of a large domain is subject to a sublinear analysis, surprisingly, the origins of this field have nothing to do with "big data" or computational challenges in data analysis. The birth of sublinear algorithms is in computational complexity theory [41]. Hence, practical performances of these methods on real applications have not been fully investigated. Recent work by some of the authors showcase the potential of sampling algorithms in graph analysis [45, 46, 29, 25, 24, 3], and the generation of application-independent generation of colormaps [50].

[1] $f(n) = o(g(n))$ means for all $c > 0$ there exists some $k > 0$ such that $0 \leq f(n) < cg(n)$ for all $n \geq k$. The value of k must not depend on n, but may depend on c.

Potential of Sublinear Algorithms in Mitigating HPC Challenges Moving analysis algorithms in-situ poses a number of technical challenges. First, algorithms will be sharing compute resources with the simulation, which poses constraints on the algorithmic choices. In particular, memory is expected to be a bottleneck in exascale computers and beyond (see Table 1.1), and thus we may have to work with data structures of the application, and not be able to build auxiliary data structures that will improve the performance of our algorithms. Secondly, the data layout will be dictated by the simulation, locally at the node level and globally at the system level. This layout will not necessarily be favorable for our algorithms, yet pre-processing to move the data will be infeasible at large scales. Thirdly, algorithms performed in-situ need to be fast, so that they do not slow down the simulation computation. For instance, any analysis to enable judicious I/O cannot take more time than the I/O itself.

In addition to making analysis algorithms themselves more efficient, we expect that long-term, sampling-based algorithms, especially sublinear algorithms, will also play an important role in the enabling of adaptive workflows in the exascale era and beyond. First, the small number of samples grant runtime efficiency, which enable working concurrently with the simulation, with negligible effect on runtime. The error/confidence bounds quantify the compromise in accuracy compared to full analysis. Moreover, for most problems, the number of samples required only depend on error/confidence bounds, which lead to perfect scalability of these algorithms for extreme problem sizes. The memory requirements of sublinear algorithms are also small, and typically only in the order of the samples. In some cases, additional data structures may be necessary to enable random sampling, but even such structures are not memory-intensive.

With this LDRD research, we showcase several advancements in which applications of sublinear algorithms mitigate HPC challenges posed by the paradigm shift to in-situ workflows. In Chapter 2 we show how sublinear algorithms can be deployed to identify features of interest through efficient color map generation. In Chapter 3 we show how sublinear trigger detection can be used to enable adaptive in-situ workflows. We conclude this report with a discussion of the LDRD accomplishments in Chapter 4 and a presentation of conclusions and lessons learned in Chapter 5.

Chapter 2

Identifying Features of Interest: Sublinear Colormaps[1]

> Color is the most relative medium in art.
> — Josef Albers, *Interaction of Color*

Color is one of the most prevalent tools used in scientific visualization and possibly the easiest to misuse – knowingly, or, as this work considers, unknowingly. As Albers notes, and cognitive neuroscience has established experimentally, human perception of color is only relative to its surroundings. Within a single image, color can be used to identify spatial trends at varying scales, discriminate between neighboring values, and even gauge absolute values to some degree.

However, there are very few tools to design the maps between numbers we wish to illustrate and colors that will aid in their undistorted perception[2]. General-purpose visualization tools are often given data with no description of its source, no units of measurement, no accuracy of its computation or measurement, no measurements of its trends, nor any indication of how importance is defined. Large datasets that cannot be held in memory (or require a high-performance computer with distributed memory) may not provide easy access to such summary information. It is difficult to choose a good colormap with some expert knowledge of the dataset.

Most default colormaps are defined by linearly interpolating dataset values between the maximum and minimum in the range of the dataset. This is computationally cheap but disregards the distribution of values within the data. Consider Figures 2.3(a) and 2.3(b), which show temperatures inside a rotating disk reactor. These were generated by the standard tools VisIt and ParaView respectively. It is not all clear from these figures that there are three special boundary conditions in this data. For example, the entire outer surface of the cylinder has the same temperature of 293.15°C, distinct from all other points. The bottom part has a temperature of 303.15°C, also distinct from all other points. These are completely obscured by the simplistic linear interpolation of color.

[1] The material presented in this chapter was also reported in [50].

[2] It is arguably impossible to say that a particular person's perception is biased or unbiased, but it may be possible to quantify how a population's perception of a particular feature is proportional to the evidence for it provided by the data *relative to other features in the same data*.

Clearly, choosing an informative colormap requires some data analysis. But analysis techniques providing detailed information about the data require possibly expensive preprocessing stages. For instance, information regarding relative frequencies of values and spatial relationships could be useful for a good colormap. But how does one obtain this information for a large dataset without affecting the running time of the actual visual presentation? In this work, we address this problem with an efficient method to generate colormaps via random sampling. These colormaps identify discrete values that occur with high probability – some fraction τ of the dataset or larger – and apply a continuous color-palette curve to the remaining CDF by quantile, so that perceptible changes in color are assigned to equiprobable ranges of values.

Theoretically, our algorithm is simple and provably robust. In practice, with a negligible overhead our algorithm yields images that better highlight features within the data. Most importantly, the required sample size depends only on desired accuracy parameters and is independent of the dataset size. However, as with any sampling approach, this technique may fail to discover exceedingly rare events; we relate sample size to the probability $1-\delta$ that an event more frequent than τ goes undetected. Events less frequent than τ are considered negligible. For reasonably small values of τ and δ, the sample size is practical – but it does grow quickly as τ is decreased.

Contributions

Consider visualizing a large dataset. If there are certain "prominent" values that occur with high frequency, then we might want these to take on special colors to highlight them. Continuous data may not uniformly occupy its range, and for some applications it is important to visualize overall trends in the colored attribute while in others it is small deviations from the trend that are important. We devise a new simple and scalable algorithm to design such color maps. The salient features of our algorithm follow.

- We introduce a simple sampling-based routine for approximately identifying prominent values and ranges in data. We provide a formal proof of correctness (including quantifiable error bounds) for this routine using probability concentration inequalities.
- The number of samples required by our routine only depends on the accuracy desired and *not on the data size*. For example, suppose we wish to find all values that occur with more than 1% frequency (in the data). Then, the number of samples required is some fixed constant, regardless of data size.
- Given these approximate prominent values and ranges in the data, we provide an automated technique for generating discrete and/or continuous color maps.

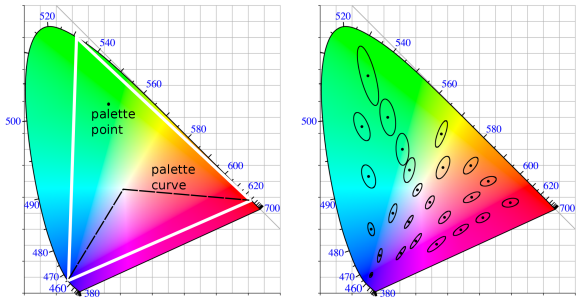
We empirically demonstrate our results on a variety of datasets. Consider Figure 2.3. The rightmost figure shows the coloring output by our algorithm, and notice how it picks up the three prominent values (and one range) in the data. This allows for a coloring that highlights the boundary conditions, as opposed to the standard colormaps.

2.1 Related Work and Background

The most well-known work on color in scientific visualization is Brewer's treatise on the selection of color palettes [13, 14, 15]. Her thesis and much surrounding literature [39, 26] focus on choosing palettes that 1) avoid confounding intensity, lightness, or saturation with hue either by co-varying them or using them to convey separate information; and 2) relate perceptual progressions of color with progressions of values in the data to be illustrated or – when the values being illustrated have no relationship to each other – to avoid perceptual progressions of colors so that the image does not imply a trend that is absent in the data. Other work[34, 11] discusses fundamental flaws with the commonly used default "rainbow" colormap and presents results on diverging color maps which have since been adopted by many in the community as they perform much better than the rainbow colormap in scientific visualization settings. Eisemann et al. [18] propose a family of pre-color-map transforms that transition from linear scaling (where all values in a range have the same importance) to a kind of histogram-equalized scaling (where values that frequently occur in the dataset are given more importance). They claim that linear scaling provides a way to discover outliers, but this is only true if the data has a single central tendency; outliers between multiple tendencies could well be masked by a linear scale. Also, perceptual differences between colors mapped to values are not considered. Finally the technique is not inherently scalable since it requires sorting all observed values, although it could likely be adapted to use representative subsamples. However, they identify a key factor in colormap design for exploratory visualization: without problem-specific knowledge, attempts to improve discrimination between values oppose attempts to remove unused ranges of values.

Other significant work has studied the generation of transfer functions for volume rendering. Here, work includes the use of entropy to maximize the "surprisal" and thus the information content of the image [9].The work of [38, 27] compare a number of transfer function generation techniques and provide good high-level overviews of the research in this area. Borkin et al. [10] note that when a specific setting is being targeted, visualizations – including the colormap – should be chosen to match. However, in this work we consider the task given to general-purpose visualization tools: how should default renderings of datasets provided without any context be created? Finally, tone mapping [30] has been used to adjust images that contain more contrast than their presentation medium is able to provide by modeling how the human visual system deals with contrast.

A *color model* is a mathematical abstraction in which colors are represented as tuples of values. Common color models include RGB, CMYK, and HSV. These and other color models differ in how the tuples of values are interpreted and combined to achieve the spectrum of attainable color values. For example, RGB uses additive color mixing, storing values of red, green, and blue. Not all devices can represent and capture colors equally. A *color space* defines a relationship between coordinates defined by a color model and the human perception of those coordinates – over some subset of coordinates that may be perceived. The *gamut* of a device is defined to be the subset of the color space that can be represented by the device, and those colors that cannot be expressed within a color model are considered out of gamut. CIELAB is a color space that was created to serve as a device-independent



(a) (b)

Figure 2.1. The horse-shoe shape in these figures demonstrates the chromaticities visible to the average human. In (a) the white triangle is the gamut of the RGB color space and a palette curve and palette point are shown in this space. In (b) multiple MacAdam ellipses are shown to highlight perceptual non-uniformities in the color space.

model to be used as a reference. It describes all colors that the human eye can see and is defined by three coordinates: L represents the lightness of a color, a represents its position between green and magenta/red, and b represents its position between yellow and blue.

The distance between colors is often defined in terms of the Euclidean distance between two colors in CIELAB space and is typically referred to as ΔE. Different studies have proposed different ΔE values that have a *just noticeable difference (JND)*. Often a value of ΔE≈ 2.3 is used, however, several variants on the ΔE function have been introduced to address perceptual non-uniformities in the CIELAB color space. These non-uniformities can be visually depicted by *MacAdam ellipses*, which are elliptical regions that contain colors that are considered indistinguishable. These ellipses were identified empirically by MacAdam [32] who found that the size and orientation of ellipses vary widely depending on the test color. Figures 2.1(a) and 2.1(b) demonstrate the *chromaticities* (the quality of a color independent of its brightness), visible to the average human. In Figure 2.1(a) the white triangle is the gamut of the RGB color space and a *palette curve* and *palette point* are shown in this space. In Figure 2.1(b) multiple MacAdam ellipses are shown to highlight perceptual non-uniformities in the color space.

2.2 Overview of Our Approach

It is convenient to think of a dataset as a distribution $\mathcal{D}$ of values. Formally, pick a uniform random point in the data, and output the value at that point. This induces the distribution $\mathcal{D}$ on values we focus upon. We will fix $\tau \in (0, 1)$ and positive integer ν as parameters to

our procedure. We will set $\delta \in (0, 1)$ as a failure probability for our algorithm.

We begin with identifying *prominent values*. These are values that make up at least a τ-fraction of the complete dataset. (In terms of $\mathcal{D}$, these are values with at least τ probability.) Once these are identified, these can be "removed" from $\mathcal{D}$ to get a distribution $\mathcal{D}'$. This can be viewed as modeling $\mathcal{D}$ with a mixture of a discrete distribution and another distribution $\mathcal{D}'$ that we approximate as continuous. Formally, $\mathcal{D}'$ is the distribution induced by $\mathcal{D}$ on all values except the prominent ones.

Next we divide the real line into ν intervals, where each interval has $1/\nu$ probability in $\mathcal{D}'$. This splitting provides an approximate CDF – to within $1/\nu$ with probability $1-\delta$ – that can be used for coloring data other than prominent values by quantile (i.e., histogram-equalized).

One of our contributions is a simple algorithm that (provably) approximately computes this in time that only depends on τ, ν, δ. It does not depend on the size of the actual data, and only on the required precision (which is quantified by these parameters).

To generate our colormap, we first assign the prominent values perceptually distant palette points. Intervals of equal probability are then distributed evenly along a palette curve parameterized by perceptual uniformity in order to make the distribution of data over the scale as perceptible as possible. This gives the final colormap. Finally, we provide a second colormap that alternates luminance between light and dark values for each equiprobable portion of $\mathcal{D}'$ in order to aid in identifying small local deviations within a single tendency of the data.

The interval identification is thus effectively a sample-based approximation to histogram equalization, however our algorithm performs this step *after* the prominent values have been separated from the dataset samples so that discrete behavior will not bias the density estimate [3].

Because quantiles are less sensitive to outliers than the PDF, extreme values do not have an exaggerated effect on coloring. Our approach does not provide outlier identification and we contend that – for large data – outliers should be considered in a separate sampling biased toward them once prominent intervals have been confirmed as conceptually significant by a domain expert. Otherwise small samples are unlikely to be effective at identifying outliers.

It is important to note that sampling is important in order to obtain scalability [37]; when data is distributed across multiple processes with no guarantees on the uniformity of the distribution. A naive exact CDF computation can easily exceed the memory available to a single process, while bucketing can require significant memory and network bandwidth to properly compute.

[3] One example of mixed discrete and continuous behavior is rainfall totals; samples are generally modeled [49] as a mixture [21] of clear days (a discrete distribution with only 1 possible rainfall total) and rainy days (which tend to have an exponential or, more generally, gamma distribution of precipitation totals [23]). One might expect – and this work demonstrates – similar behavior from simulations, where some regions are static due to boundary or initial conditions while others evolve into approximations of continuous behavior.

The next section details the algorithm we use for computing colormaps; after that, we present the mathematics, both proof and algorithms, for sampling and for detecting prominent values and important intervals.

2.3 Mapping Scalar Values to Color

We formally describe our procedure for mapping data values to color according to the distribution of scalar values within the data. We assume the following are provided as input to the mapping algorithm: 1) a palette curve and palette points; and 2) a list of prominent values in the data and an approximate CDF for the remaining data. The CDF comprises a collection of n disjoint and contiguous intervals, $B_1, \ldots, B_n$. Each B_i has an associated minimum function value f_i^{min} and maximum function value f_i^{max}, simply corresponding to the left and right endpoints of B_i. Furthermore, each B_i also has an associated set of samples, whose size is denoted by $s(B_i)$. See the bottom of Figure 2.2(b), where the relative heights of bars in B_i denote the size of $s(B_i)$. These represent an approximate CDF in that (roughly speaking), the CDF value at the right endpoint of B_i is given by $\sum_{j=1}^{i} s(B_j) / \sum_{j=1}^{n} s(B_j)$. (In Section 2.4, we describe a provably robust sampling-based approach for the quick estimation of this information.)

Given the input, we first assign a unique color to each prominent value. Next, we discretize the palette curve p into k individual palette points of $\Delta\text{E} \approx 2.3$ (this value is tunable), see Figure 2.2(b).

Given a scalar value, f we compute an interpolation factor, t_f, based on the position of f in the CDF. (Again, refer to Figure 2.2(b).) To do this we identify the bucket B_i that contains f and compute:

$$t_f = \frac{\sum_{j=1}^{i-1} s(B_j) + \left(\frac{f - f_i^{min}}{f_i^{max} - f_i^{min}}\right) * s(B_i)}{\sum_{j=1}^{n} s(B_j)}$$

Once we have identified t_f, we compute the final color, c_f, using the palette curve p with k palette points as

$$\begin{aligned} j &= t_f * k, \\ t_j &= j - \texttt{floor}(j), \\ c_f &= c_j + t_j(c_{j+1} - c_j). \end{aligned}$$

Implementation Details: As Eisemann et al. note [18], colormap definitions have opposing objectives in exploratory visualization where problem specifics are unknown; it is impossible to discern whether similar values should be perceptually similar in order that

trends across large differences in function values may be detected or whether similar values should be perceptually distinct in order that small differences may be detected. We provide two colormaps for these two situations that users must choose between: an *inter-mode* map that varies hues smoothly and at constant luminance within CDF intervals and an *intra-mode* colormap that varies hue smoothly but rapidly alternates luminance between high and low values to provide visual cues for discriminating between small differences in value. We name the situations inter- and intra-mode because one might find the former useful for contrasting behavior across different spatial or statistical modes; and the latter useful for contrasting behavior within a given mode. This use of luminance is similar to the use of structured light to highlight small geometric features [52]. Furthermore, prominent values can be emphasized or de-emphasized within an image by modifying the lightness of the associated color in CIELAB space. We use the Little CMS color engine [33] to perform all transformations between color spaces and compute ΔE distances between colors.

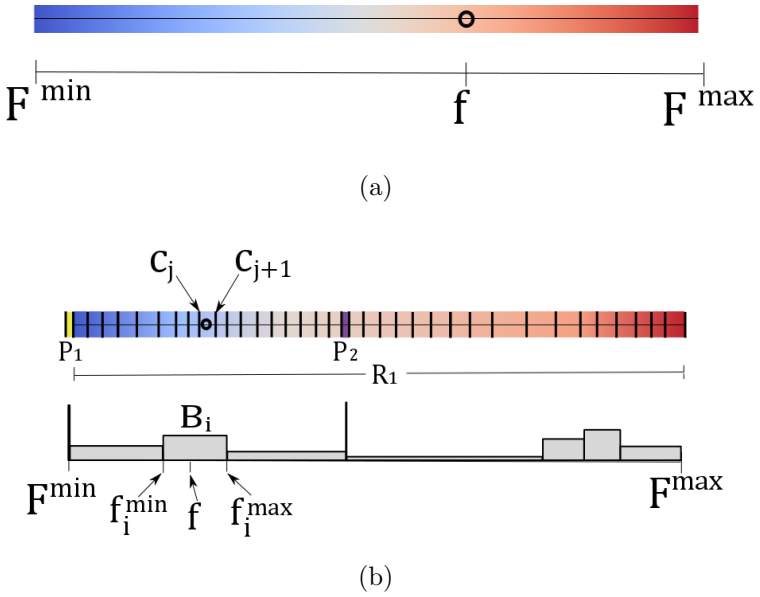


Figure 2.2. Consider a dataset with two prominent values, P_1, P_2 and overall range R_1. In figure (a) a simple linear interpolation scheme is depicted in which the color associated with the value f is independent of the distribution of values in the dataset. In (b) the two prominent values are assigned the colors yellow and purple respectively. The range is assigned blue-red. Using our CDF-based interpolation scheme, we obtain a color for the function value f that is determined by the important interval and associated range to which f belongs. In out future work, we point out that should we have a way to split ranges, a second green-orange palette curves might be used to illustrate samples from apparently distinct populations.

2.4 Prominent Values and the CDF Approximation

We describe the sampling approaches used to determine prominent values and important intervals of a dataset. We employ two simple sampling algorithms for this purpose. The first algorithm does a direct sampling to determine frequent values in the dataset. The second algorithm constructs a series of intervals, such that the frequency of data within each interval is roughly the same. Note that a large variance in the lengths of these intervals indicates a non-uniformity in data value distribution. These intervals represent our approximate CDF.

We treat our data as a *discrete* distribution, where for each value r, p_r is the fraction of the dataset where the value r is attained (So $\{p_r\}$ describes a distribution over the range of the dataset). We use $\mathcal{D}$ to denote this distribution and R to denote the support. For any set S (often an interval of the real line), we use $P(S)$ to denote the probability mass of S.

The analysis of both algorithms follow from straightforward applications of Chernoff bounds. We state the *multiplicative Chernoff bound* (refer to Theorem 1.1 in [17]) for sums of independent random variables.

Theorem 2.4.1 *[Chernoff bound] Let $X_1, X_2, \ldots, X_k$ be independent random variables in $[0,1]$ and $X = \sum_{i=1}^{k} X_i$.*

- *(Lower tail) For any $\varepsilon > 0$,*

$$\Pr[X < (1-\varepsilon)\mathbf{E}[X]] \leq \exp(-\varepsilon^2 \mathbf{E}[X]/2).$$

- *(Upper tail) For any $\varepsilon > 0$,*

$$\Pr[X > (1+\varepsilon)\mathbf{E}[X]] \leq \exp(-\varepsilon^2 \mathbf{E}[X]/3).$$

- *(Upper tail) For any $t > 2e\mathbf{E}[X]$,*

$$\Pr[X > t] \leq 2^{-t}.$$

In our theorems, we do not attempt to optimize constants. The running time of our algorithms does not depend on the data size, and the theorems are basically proof of concepts. Our empirical work will show that the actual samples sizes required are quite small. For convenience, we use c and c' to denote sufficiently large constants. Our algorithms will take as input a sample size parameter. Our theorems will show that this can be set to a number that only depends on precision parameters, for desired guarantees.

2.4.1 Finding Prominent Values

Our aim is to determine values of r such that $p_r > \tau$ is large, where $\tau \in (0,1)$ is a *threshold parameter*.

`find-prominent`(s, τ)

Inputs: sample size s, threshold τ
Output: the "frequent set" of range elements, I.

1. Generate set S of s independent random samples from $\mathcal{D}$
2. Initialize important set $I = \emptyset$.
3. For any element $r \in \mathcal{D}$ that occurs more than $s\tau/2$ times in S,
   Add r to I.
4. Output I.

The following theorem states that (up to some approximation), I is indeed the set of frequent elements. The constants in the following are mainly chosen for presentation. (Instead of $p_r < \tau/8$ in the following, we can set it to τ/α, for any $\alpha > 1$, and choose s accordingly.) Throughout our theorems, we use δ for a tunable error parameter that decides the sample size s. Note that it is not an explicit parameter to the algorithms.

Theorem 2.4.2 *Set $s = (c/\tau)\ln(c/(\tau\delta))$. With probability $> 1 - \delta$ (over the samples), the output of* `find-prominent`(τ, δ) *satisfies the following.*
If $p_r > \tau$, then $r \in I$.
If $p_r < \tau/8$, then $r \notin I$.

Proof: We first prove that with probability at least $1 - \delta/2$, for all $p_r > \tau$, $r \in I$. Then we show that with probability at least $1 - \delta/2$, for all $p_r < \tau/8$, $r \notin I$. A union bound then completes the proof.

Consider some r such that $p_r > \tau$. Let X_i be the indicator random variable for the event that the ith sample is r. So $\mathbf{E}[X_i] = p_r$ and all X_is are independent. We set $X = \sum_{i \le s} X_i$ and apply the Chernoff lower tail of Theorem 2.4.1 with $\varepsilon = 1/2$. Hence, $\Pr[X < \mathbf{E}[X]/2] \le \exp(-\mathbf{E}[X]/8)$. Note that $\mathbf{E}[X] = p_r s > s\tau$. We obtain $\Pr[X < s\tau/2] \le \Pr[X < \mathbf{E}[X]/2] \le \exp(-\mathbf{E}[X]/8) \le \exp(-s\tau/8)$. Since $s = (c/\tau)\ln(c/(\tau\delta))$, $\exp(-s\tau/8) = \exp(-c\ln(c/(\tau\delta)) \le \delta\tau/16$.

Putting it together, $\Pr[X < s\tau/2] < \delta\tau/16$. Hence, in our algorithm, the element r will *not* be in I with probability at most $\delta\tau/16$. There are at most $1/\tau$ values of r such that $p_r > \tau$. By the union bound, the probability that there exists some such value of r occurring less than $s\tau/2$ times is at most $\delta/16$. Hence, with probability $> 1 - \delta/16$, $\forall p_r > \tau$, $r \in I$.

Define set $A = \{r | p_r \ge \tau/8\}$, and $R' = R \setminus A$. The second part is stated as Lem. 2.4.3 below with $\alpha = \tau/8$. We get that with probability $> 1 - \delta/2$, all $r \in R'$ individually occur less than $s\tau/2$ times. Hence, none of them are in I.

Lemma 2.4.3 *Let $\alpha \in (0, 1)$ and $s > (c/8\alpha)\ln(c/(8\alpha\delta))$. Consider a set R' such that $\forall r \in R'$, $0 < p_r \le \alpha$. With probability $> 1 - \delta/2$, the following holds. For all $r \in R'$, the number of occurrences of r in s uniform random samples from $\mathcal{D}$ is at most $4s\alpha$.*

Proof: We apply Claim 2.4.4 (given below). This gives a series of intervals $R_1, R_2, \ldots, R_n$, such that for all $m < n$, $P(R_m \cap R') \in [\alpha, 2\alpha]$. Also, $P(R_n \cap R') \leq 2\alpha$.

Consider some R_m for $m < n$, and let Y_i be the indicator random variable for the ith sample falling in R_m. We have $\mathbf{E}[Y_i] \in [\alpha, 2\alpha]$ and $\mathbf{E}[Y] \in [\alpha s, 2\alpha s]$ (where $Y = \sum_{i \leq s} Y_i$). It will be convenient to use the bound $\mathbf{E}[Y_i] \in [\alpha/2, 2\alpha]$ and $\mathbf{E}[Y] \in [\alpha/2s, 2\alpha s]$.

Since the Y_is are independent, we can apply the first Chernoff upper tail with $\varepsilon = 1$. This yields $\Pr[Y > 2\mathbf{E}[Y]] \leq \exp(-\mathbf{E}[Y]/3)$. Since $\mathbf{E}[Y] \leq 2\alpha s$, $\Pr[Y > 4s\alpha] \leq \Pr[Y > 2\mathbf{E}[Y]]$. Since $\mathbf{E}[Y] \geq \alpha s/2$, $\exp(-\mathbf{E}[Y]/3) \leq \exp(-\alpha s/6) = \delta\alpha/3$. Putting it together, $\Pr[Y > 4s\alpha] < \delta\alpha/3$. Hence, $\Pr[Y > 4s\alpha] \leq \exp(-\alpha s/3) = \delta\alpha/3$.

Now focus on R_n and define Y analogous to above. If $P(R_n \cap R') > \alpha/2$, we can apply the previous argument with $\varepsilon = 1$. Again, we deduce that $\Pr[Y > 4s\alpha] \leq \exp(-s\alpha/6) = \delta\alpha/3$. If $P(R_n \cap R') < \alpha/2$, we apply the second Chernoff tail with $t = 4s\alpha$ (observing that $4s\alpha \geq (2e)\alpha/2$) to deduce that $\Pr[Y > 4s\alpha] \leq 2^{-4s\alpha} \leq \delta\alpha/3$.

We apply the union bound over all R_m for $m \leq n$. Note that n is at most $1/\alpha + 1$, since $P(R_m \cap R') \geq 2\alpha$ and the R_ms are disjoint. So with probability at most $\delta/2$, there exists some R_d such that number of occurrences in R_d is more than $4s\alpha$. Hence, with probability at least $1 - \delta/2$, no element in R' can appear more than $4s\alpha$ times.

Claim 2.4.4 *Let $\alpha \in (0, 1)$. Consider a set R' such that $\forall r \in R'$, $0 < p_r \leq \alpha$. There exists a sequence of numbers $\min_{r \in R'} r = z_1, z_2, \ldots, z_k = \max_{r \in R'} r$ ($k \geq 2$) such that for all $i < k - 1$, $P([z_i, z_{i+1}) \cap R') \in [\alpha, 2\alpha]$ and $P([z_{k-1}, z_k]) \leq 2\alpha$.*

Proof: This is done through a simple iterative procedure. We start with $z_1 = \min_{r \in R'} r$. Given z_i, we describe how to find z_{i+1}. Imagine z_{i+1} initialized to z_i and continuously increased until the $P([z_i, z_{i+1}] \cap R')$ (note that we use a closed interval) exceeds 2α. If z_{i+1} crosses $\max_{r \in R'} r$, then we have found the last interval and terminate this process. Now, $P([z_i, z_{i+1}) \cap R')$ (the open interval) must be less than 2α, or we would have stopped earlier. Furthermore, $P([z_i, z_{i+1}) \cap R') = P([z_i, z_{i+1}] \cap R') - p_{z_{i+1}} \geq 2\alpha - \alpha = \alpha$.

2.4.2 Finding an Approximate CDF

Our aim is to construct a series of disjoint intervals that (almost) equally partition the probability mass. To gain some intuition, consider a positive integer v and a sequence of numbers $y_0, y_1, \ldots, y_v$ where $y_0 = \min_{r \in R} r$, $y_v = \max_{r \in R} r$, and for all $i < v$, $P([y_i, y_{i+1})) = 1/v$. Our algorithm will try to find these intervals (for a parameter v). Of course, such intervals may not even exist, due to the discrete nature of $\mathcal{D}$. Nonetheless, we will try to find suitable approximations. We assume that there are no values in $\mathcal{D}$ with high probability. This is an acceptable assumption, since we run this procedure after "removing" prominent values from $\mathcal{D}$.

There are two parameters for `find-CDF`: the sample size s and the block size b. For convenience, assume b divides s. The output is a series of s/b intervals, each with (provably) approximately the same probability mass. As we mentioned earlier, this constitutes an approximation to the CDF, since the probability mass of the first k of these intervals will be (approximately) proportional to k.

`find-CDF`(s, b)
Inputs: sample size s, block size b
Outputs: Intervals $B_1, B_2, \ldots$

1. Generate set S of s independent random samples from $\mathcal{D}$.
2. Sort these to get the (ordered) list $\{x_1, x_2, x_3, \ldots, x_s\}$.
3. Output the intervals $B_1 = [x_1, x_b)$, $B_2 = [x_{b+1}, x_{2b})$, etc. In general, the ith interval B_i is $[x_{(i-1)b+1}, x_{ib})$ and there are s/b blocks. The samples in this interval form the associated set, so $s(B_i) = |B_i \cap S|$.

This main theorem involves some play of parameters, and we express s and b in terms of an auxiliary (integer) parameter v. Again, the constants chosen here are mainly given for some concreteness and notational convenience. We use the notation $A \in (1 \pm \beta)B$ as a shorthand for $A \in [(1-\beta)B, (1+\beta)B]$.

Theorem 2.4.5 *Set $s = cv\ln(cv/\delta)$ and $b = s/v$. Suppose there exists no $r \in R$ such that $p_r > 1/100v$. With probability $> 1 - \delta$, the following holds. For each output interval B, $P(B) \in (1 \pm 1/10)/v$. Furthermore, $P((\min_{r\in R} r, x_1))$ and $P((x_s, \max_{r\in R} r))$ are at most $1/50v$.*

Proof: We first apply Claim 2.4.4 with $\alpha = 1/100v$ and $R' = R$, to get the sequence $\min_{r\in R} r = z_1, z_2, \ldots, z_k = \max_{r\in R} r$. We have $P([z_i, z_{i+1})) \in [1/100v, 1/50v]$ for $i < k-1$ and $P([z_{k-1}, z_k]) \leq 1/50v$. We prove the following lemma.

Lemma 2.4.6 *Set $s = cv\ln(cv/\delta)$. Let $Z_{i,j}$ be the number of samples falling in interval $[z_i, z_j)$. With probability at least $1 - \delta/2$, for all pairs i, j, $Z_{i,j} \in (1 \pm 1/20)sP([z_i, z_j))$.*

Proof: Fix interval $[z_i, z_j)$. Let X_m be the probability of the mth sample falling in $[z_i, z_j)$. We have $\mathbf{E}[X_m] = P([z_i, z_j)) \geq 1/100v$ and all X_m's are independent. Also, $Z_{i,j} = \sum_{m\leq s} X_m$, and $\mathbf{E}[Z_{i,j}] = sP([z_i, z_j)) \geq s/100v$. The upper Chernoff tail with $\varepsilon = 1/20$ yields $\Pr[Z_{i,j} < (1 - 1/20)sP([z_i, z_j))] \leq \exp(-sP([z_i, z_j))/800)$. The lower tail with $\varepsilon = 1/20$ yields $\Pr[Z_{i,j} > (1 + 1/20)sP([z_i, z_j))] \leq \exp(-sP([z_i, z_j))/1200)$.

By a union bound over both tails and plugging in the bound $P([z_i, z_j)) \geq 1/100v$, the probability that $X_{i,j} \notin (1 \pm 1/20)sP([z_i, z_j))$ is $\exp(-s/(800 \cdot 100v)) + \exp(-s/(1200 \cdot 100v))$. Doing the calculations (for sufficiently large constant c), this probability is at most δ/cv^2. A union bound over all intervals (at most $\binom{100v+1}{2}$ of them) completes the proof.

We apply Lem. 2.4.6, so with probability $1-\delta/2$, for all i, j, $Z_{i,j} \in (1 \pm 1/20)sP([z_i, z_j])$. Now, we apply Lem. 2.4.3 with $\alpha = 1/100v$ and $R' = R$. With probability $> 1-\delta/2$, no element occurs more than $s/25v$ times. By a union bound, both these hold with probability $> 1-\delta$. Under these conditions, we complete our proof.

Fix a block B, and let $[z_i, z_j)$ be the smallest such interval that contains B. Let $[z_{i'}, z_{j'})$ be the largest such interval inside B. Since (for any a) $P([z_a, z_{a+1})) \leq 1/50v$, $P([z_i, z_j)) - 2 \cdot (1/50v) \leq P(B) \leq P([z_i, z_j))$. Similarly, $P([z_{i'}, z_{j'})) \leq P(B) \leq P([z_{i'}, z_{j'})) + 2 \cdot (1/50v)$.

Let ℓ denote the number of sample points in $B = [x_h, x_{h+1})$. Since the number of samples in $[x_h, x_{h+1}]$ (the closed interval) is exactly b, $\ell \leq b = s/v$. Also, ℓ is at least b minus the number of sample occurrences of x_{h+1}. So $\ell \geq b - s/25v = (1-1/25)s/v$. We remind the reader that $Z_{i,j}$ is the number of sample occurrences in $[z_i, z_j)$, which is an interval containing B. Hence $\ell \leq Z_{i,j}$. Similarly, $\ell \geq Z_{i',j'}$.

We use the bound that $Z_{i,j} \in (1 \pm 1/20)sP([z_i, z_j))$ (similarly for $Z_{i',j'}$). Since $\ell \leq X_{i,j}$, we get $(1-1/25)s/v \leq (1+1/20)sP([z_i, z_j))$. Since $\ell \geq X_{i',j'}$, $s/v \geq (1-1/20)sP([z_{i'}, z_{j'})$. Relating the probabilities to $P(B)$, we get $(1-1/25)/v \leq (1+1/20)(P(B)+1/25v)$ and $1/v \geq (1-1/20)(P(B)-1/25v)$. Rearranging the terms, we complete the proof for output interval B.

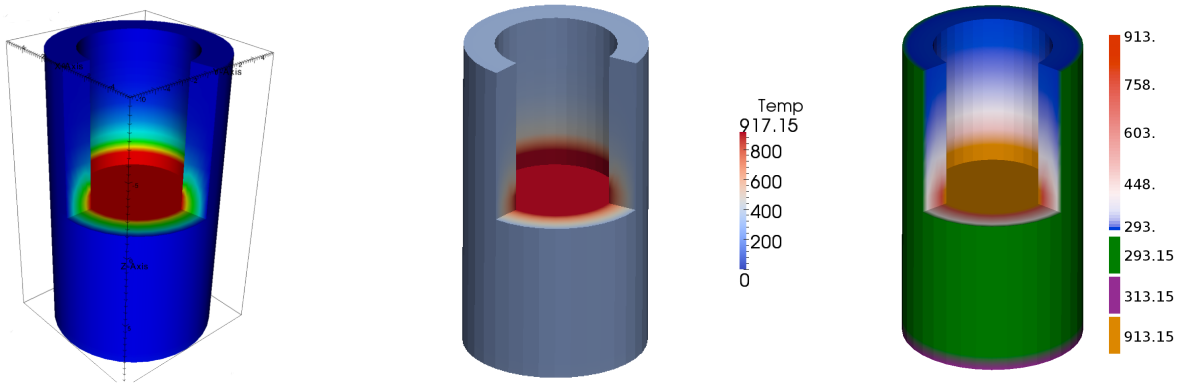
No samples fall in the intervals $(\min_{r \in R} r, x_1)$ and $(x_s, \max_{r \in R} r)$. Hence, they must be completely contained in some interval of the form $[z_a, z_{a+1}]$, and their probabilities are at most $1/50v$.

2.5 Sublinear Colormap Results

In this section we demonstrate the results of applying our colormap generation algorithm to a variety of datasets. These include the Mandelbrot dataset, a rotating disk reactor, and two combustion datasets. We compare images of the data created with default colormaps of several visualization tools to colormaps created using our technique.

The Mandelbrot dataset is a synthetic function that can be sampled at any resolution. By contouring the volumetric function using a geometric sequence of isovalues, we obtained auxiliary data used to characterize how prominent value detection behaves as distinct values become near enough to appear as a continuous range, see Figure 2.6. We also run the algorithm on the data without contouring in Figure 2.7.

The rotating disk reactor, shown in Figure 2.3, is a steady-state simulation of continuous vapor deposition (CVD) carried out by MPSalsa [43, § D.2]. A notch has been removed for illustrative purposes. It is a small dataset that contains thermodynamic state variables, velocity, and chemical species concentrations at each node. The simulated region is a fluid-filled cavity between a concentric tube and rod where reactants flow up from the unobstructed volume (bottom), across the heated rod, and exit at the annulus (top). Of interest is the fact



(a) VisIt (b) ParaView (c) Prominent values

Figure 2.3. Temperature inside a rotating disk reactor. Sampling is able to identify constant boundary conditions and highlight them where they would otherwise be obscured by the full scalar value range.

that temperature boundary conditions have been imposed: the outer cylindrical wall is held at an ambient temperature of 293.15 K, the inner rod cap is heated to 913.15 K, and the fluid entering the chamber is a constant 303.15 K. Because the two lower temperature conditions are very near each other relative to the heated rod cap, it is impossible to distinguish them in the default views of ParaView and VisIt. However, by passing the temperature through the sampling algorithm above, we obtain the PDF and CDF shown in Figure 2.4, along with associated prominent values that pull out these features. By assigning colors far from the palette curve to these prominent values, the difference is apparent.

In addition to these two datasets we also demonstrate our results on two combustion datasets: HCCI, and Lifted Ethylene Jet. These datasets were generated by S3D [16], a turbulent combustion simulation code that performs first principles-based direct numerical simulations in which both turbulence and chemical kinetics introduce spatial and temporal scales spanning typically at least 5 decades. The HCCI dataset is a study of turbulent auto-ignitive mixture of Di-Methyl-Ether and air under typical Homogeneous Charge Compression Ignition (HCCI) conditions [4]. This simulation is aimed at understanding the ignition characteristics of typical bio-fuels for automotive applications and has a domain size of over 175 million grid points. The second simulation describes a lifted ethylene jet flame [53], involved in a reduced chemical mechanism for ethylene-air combustion, with a domain size of 1.1 billion grid points.

Figure 2.7 contains images comparing our technique with that of ParaView for the Mandelbrot, HCCI, and Lifted Ethylene Jet datasets. The first row demonstrates default color maps generated by ParaView for the Mandelbrot, HCCI, and Lifted Ethylene Jet datasets. The second and third row contrast inter- vs. intra-mode differences with the prominent values emphasized as colors with low lightness in CIELAB space, while the fourth and fifth

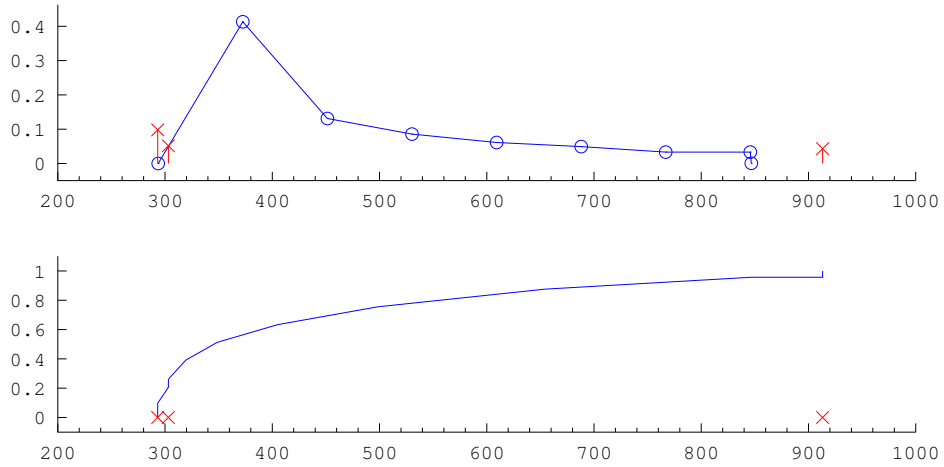


Figure 2.4. The probability density function (PDF, top) and cumulative density function (CDF, bottom) of the rotating disk reactor, shown in blue and obtained from the prominent values and blocked ranges. Prominent values and their probabilities are shown in red. Note that prominent values have an absolute probability associated with them while the PDF shows densities estimated from many different values; none of the samples used to estimate the density occur with frequencies approaching that of the prominent values.

row contrast inter- vs. intra-mode differences with prominent values de-emphasized as colors with high lightness in CIELAB space. In particular, within the combustion datasets, our approach visually depicts structures in the data that are difficult to see in the default view. These structures cover a relatively small percentage of overall scalar domain, however these values are observed with relatively high frequency within the data. We note that our approach can be effective at identifying coherent structures in spite of the fact that we are not integrating spatial information into our estimates.

Scalability: Figure 2.5 highlights the scalability of our approach for the Lifted Ethylene Jet and HCCI datasets. We performed our experiments on Lens at the Oak Ridge Leadership Computing Facility. Lens is a 77-node commodity-type Linux cluster whose primary purpose is to provide a conduit for large-scale scientific discovery via data analysis and visualization of simulation data. Lens has 45 high-memory nodes that are configured with 4 2.3 GHz AMD Opteron processors and 128 GB of memory. The remaining 32 GPU nodes are configured with 4 2.3 GHz AMD Opteron processors and 64 GB of memory. Datasets were evaluated with $\tau = 0.001$ and $b = 1024$, requiring approximately 100,000 samples to achieve a failure probability of $\delta = 10^{-6}$.

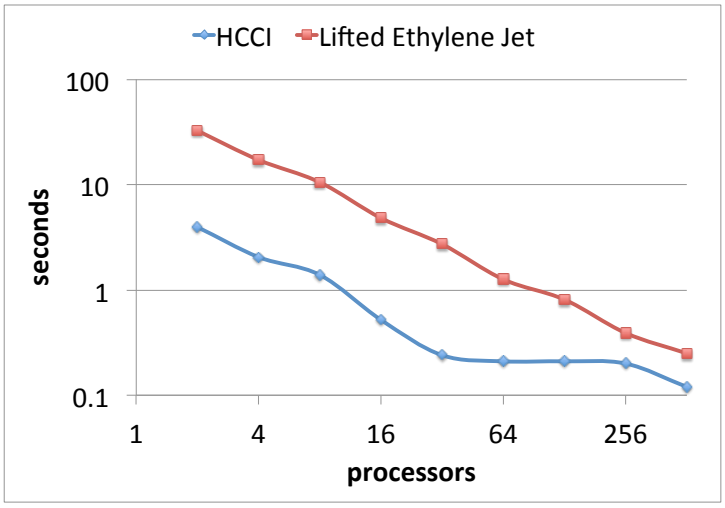
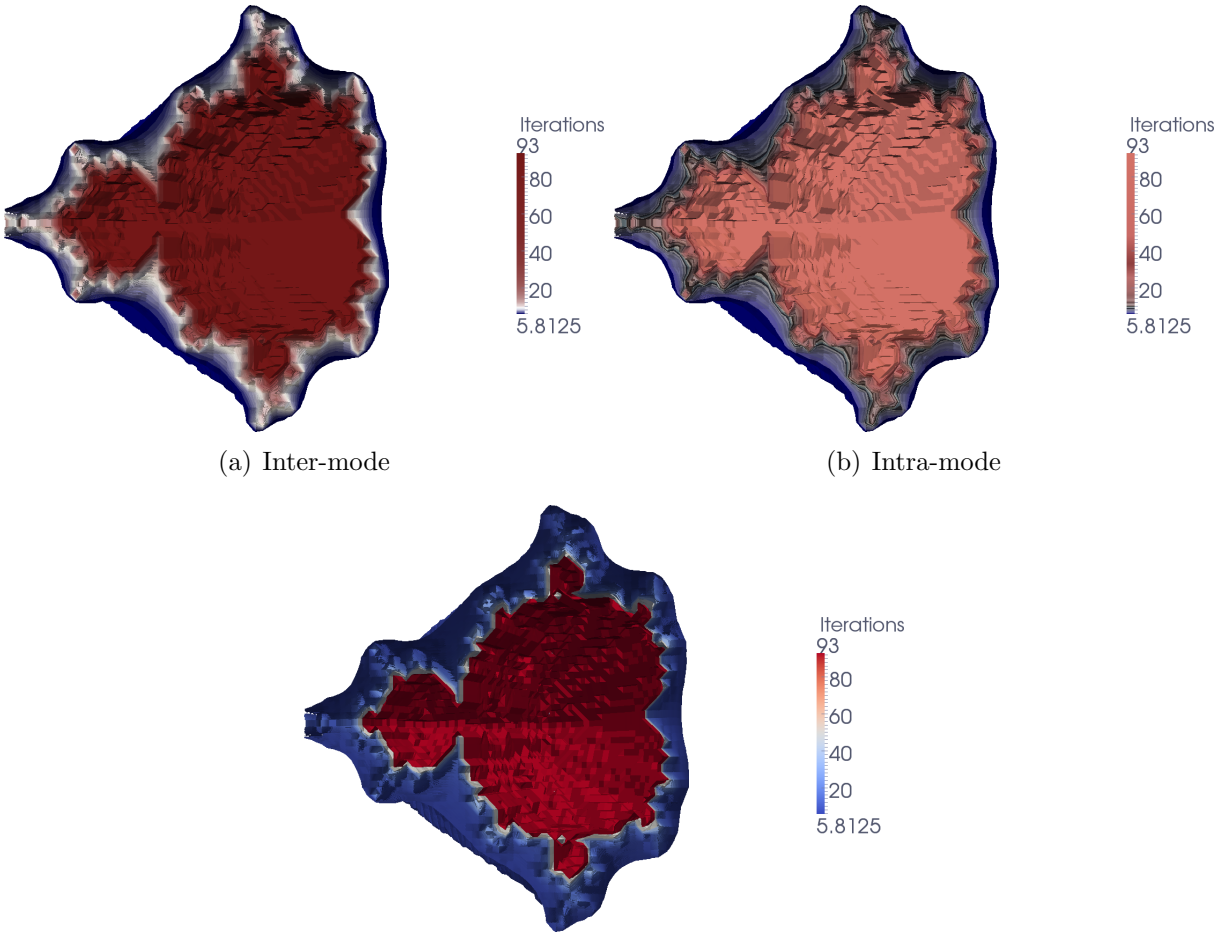


Figure 2.5. Our technique demonstrates good scalability as the number of samples required by our algorithm is fixed according to accuracy parameters as opposed to dataset size.

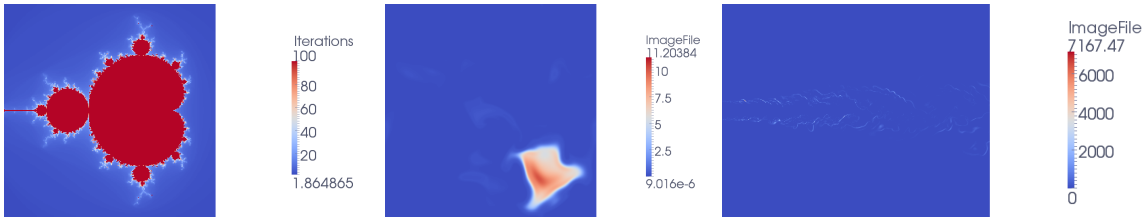


(a) Inter-mode

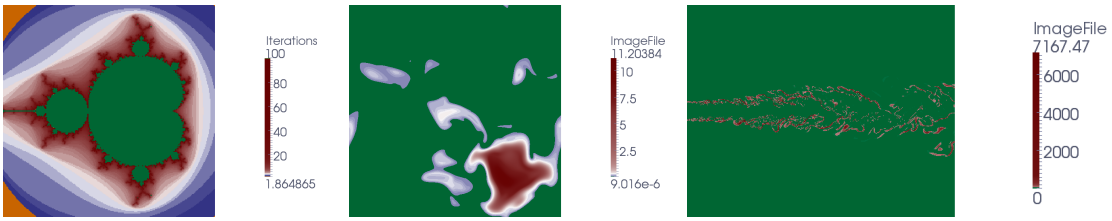
(b) Intra-mode

(c) ParaView

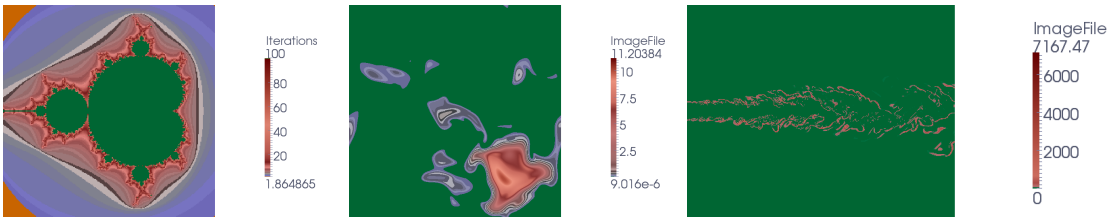
Figure 2.6. These images show 32 contour values, each of which is identified by our algorithm as a prominent value. Many of these isovalues lie closely together and are difficult to differentiate using traditional default color maps. Using our algorithm it is much easier to see that there are in fact many individual surfaces.



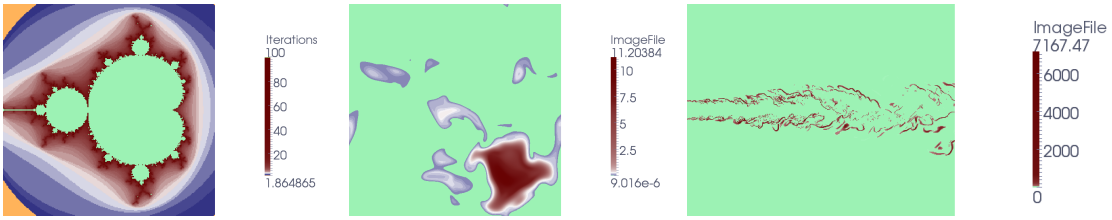
(a) ParaView Mandelbrot (b) ParaView HCCI (c) ParaView Lifted Ethylene Jet



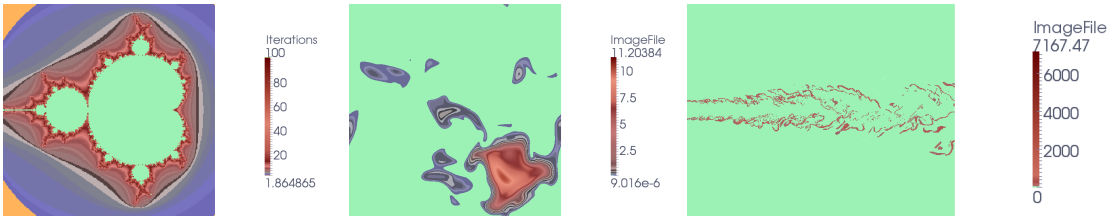
(d) Inter-mode Mandelbrot (e) Inter-mode HCCI (f) Inter-mode Lifted Ethylene Jet



(g) Intra-mode Mandelbrot (h) Intra-mode HCCI (i) Intra-mode Lifted Ethylene Jet



(j) Inter-mode Mandelbrot (k) Inter-mode HCCI (l) Inter-mode Lifted Ethylene Jet



(m) Intra-mode Mandelbrot (n) Intra-mode HCCI (o) Intra-mode Lifted Ethylene Jet

Figure 2.7. The first row demonstrates default color maps generated by ParaView for the Mandelbrot, HCCI, and Lifted Ethylene Jet datasets. The second and third row contrast inter- vs. intra-mode differences with the prominent values emphasized as colors with low lightness in CIELAB space, while the fourth and fifth row contrast inter- vs. intra-mode differences with prominent values de-emphasized as colors with high lightness in CIELAB space.

Chapter 3

Enabling Dynamic Workflows: Sublinear Trigger Detection [1]

.

In this Chapter we show how sublinear algorithms can be deployed to enable dynamic in-situ workflows, in which I/O and analysis frequencies are determined in an adaptive, data-driven manner at runtime. We introduce a methodology that is broadly applicable, yet can be specialized to provide confidence guarantees for application-specific simulations and underlying phenomena. Our methodology comprises three steps:

1. Identify a noise-resistant *indicator* that can be used to track changes in simulation state.
2. Devise a *trigger* which specifies that a property of the indicator has been met.
3. Design efficient and scalable algorithms to compute indicators and triggers.

To make decisions in a data-driven fashion, a user-defined *indicator* function must be computed and measured in-situ at a relatively regular and high-frequency. Along with the indicator, the application scientist defines an associated *trigger*, a function that returns a boolean value indicating that the indicator has met some property, for example a threshold value. Together, indicators and triggers define data-driven control-flow mechanisms within a scientific workflow, see Figure 1.1(c). While this methodology is very intuitive and conceptually quite simple, the challenges lie in defining indicators and triggers that capture the appropriate scientific information while remaining cost efficient in terms of runtime, memory footprint, and I/O requirements so that they can be deployed at the high frequency that is required.

We demonstrate how recent advances in sublinear algorithms can be used to create efficient indicators and triggers to enable data-driven control-flow mechanisms, even in those cases where standard implementations of the indicator and trigger would be significantly too expensive. Sublinear algorithms are designed to estimate properties of a function over a massive discrete domain while 1) accessing only a tiny fraction of the domain, and 2) quantifying the error or uncertainty due to using only a sample of the data. Sublinear indicators and triggers operate on a sample whose size is dependent on the accuracy of the

[1] The material presented in this chapter was also reported in [5, 44]

desired result, rather than the input size. Consequently, sublinear indicators and triggers can be deployed with *high confidence* to make workflow decisions in extreme-scale simulations. Sublinear algorithms have their limitations; in particular, they are not amenable for those control-flow decisions that are based on anomaly detection. However, they are well suited to control-flow decisions regarding general trends in data, e.g., based on quantile plots of trends of computationally expensive quantities of interest.

While our proposed approach is general, the first two steps can be made application dependent, leveraging some knowledge of the underlying physics. In this work we demonstrate our approach applied to dynamic workflow decisions in the context of direct numerical simulations of turbulent combustion using S3D [16]. In this use case, workflow decisions regarding both grid resolution and I/O frequency can be made based on the detection of a rapid increase in heat release. This means our indicator should be a precursor to heat release, and not the heat release itself. What precedes the heat release? (see Figure 3.2.) Recent studies have shown that chemical explosive mode analysis (CEMA) is a good lead indicator of heat release events [31, 47], and in this work we show that CEMA can be used to devise a noise-tolerant indicator function and trigger. CEMA is a point-wise metric computed at each grid point. Our analysis of the distribution of CEMA values shows that the range of CEMA values covered by the top percentiles, compared to the full range of CEMA values, shrinks right before heat release and then expands afterwards. This change in distribution of quantiles is illustrated in Figure 3.3 and provides the basis for our indicator and trigger.

Now that we have an intuition for what can be a good lead indicator for heat release, the next step is to devise an indicator function and associated trigger that quantify the shrinking/expansion of the portion of top quantiles of CEMA values over the full range. The challenge here is the noise tolerance. While the range of the CEMA values is defined by the minimum and the maximum, these values are, by definition, outliers of the distribution, and thus their adoption will lead to noise-sensitive triggers. Instead we use the quantiles of the distribution, which remain stable among instances of the same distribution. For instance, we can replace the minimum with the 1-percentile and the maximum with the 98-percentile, still capturing the concept of range, but yielding a much more noise-tolerant trigger, as supported by experimental results (see §3.3).

The final step is to design efficient and scalable algorithms for the indicators and triggers we have devised. Our experiments show that our CEMA-based indicators and triggers work well in practice, however, exhaustive computation of CEMA values can dominate the total simulation computation time and are thus prohibitively expensive. To overcome this computational bottleneck, we propose a sampling approach to estimate the quantiles. Our sampling based algorithm comes with provable error/confidence bounds that are only a function of the number of samples. With only 48K samples, the error will be less than %1 with confidence %99.9, which in large-scale simulation runs leads to only a few samples per processor. Most importantly, the number of samples is independent of the problem size, thus our proposed sampling algorithm offers perfect scalability. Our experiments on homogeneous charge compression ignition (HCCI) and reactivity controlled compression ignition (RCCI) simulations show that the proposed method can detect heat release, with negligible computational over-

head. Moreover our results will be used to make dynamic workflow decisions regarding data storage and mesh resolution in future combustion simulations.

The rest of the Chapter is organized as follows. §3.1 provides the background for our work; we first review the need for and the state of adaptive workflows and describe the combustion use case that is used throughout our work. §3.2 discusses the process of identifying a noise-resistant indicator and trigger for phase change in a simulation, and includes physics intuitions for how and why CEMA can be used to construct an indicator for heat release for our combustion use case. In §3.3 we demonstrate how to compute the indicator and trigger efficiently using a sublinear approach and we put all the pieces together in §3.4 to demonstrate our technique on a full scale simulation.

3.1 Background

We begin this section by reviewing recent work in enabling complex scientific computing workflows. We then provide an overview of sublinear algorithms and discuss how these mathematical techniques can be deployed to make data driven decisions in-situ. We conclude this section with a brief overview of our combustion use case.

3.1.1 Concurrent Analysis Frameworks

As we move to next generation architectures, scientists are moving away from traditional workflows in which the simulation state is saved at prescribed frequencies for post-processing analysis. There are a number of concurrent analysis frameworks available, wherein raw simulation output is processed as it is computed, decoupling the analysis from I/O. Both in-situ [54, 12, 19] and *in transit* [51, 1, 6] processing are based on performing analyses as the simulation is running, storing only the results, which are typically several orders of magnitude smaller than the raw data. This reduction mitigates the effects of limited disk bandwidth and capacity. Operations sharing primary resources of the simulation are considered in-situ, while *in transit* processing involves asynchronous data transfers to secondary resources.

Concurrent analyses are often performed at frequencies that are prescribed by the scientists *a priori*. For those analyses that are not too expensive – in terms of runtime (with respect to a simulation time step), memory footprint, and output size – the prescription of frequencies is a viable approach. However, for those analyses that are too expensive, prescribed frequencies will not suffice because the scientific phenomenon that is being simulated typically does not behave linearly (e.g., combustion, climate, astrophysics). When scientists choose a prescribed I/O or analysis frequency that is frequent enough to capture the science of interest, the costs incurred are too great, while a prescribed frequency that is cost-effective and less frequent may miss the underlying scientific effects that simulation is intended to capture. An alternative approach would be to perform expensive analyses and I/O in an adaptive fashion, driven by the data itself. In [36, 35], such techniques have been

developed based on entropy of information in the data, and building piecewise-linear fits of quantities of interest. These approaches fit within the methodology proposed here and are domain-agnostic. In this work we present a strategy that can leverage the scientists' physics intuitions, even when the in-situ analyses that captures those intuitions would otherwise be too expensive to compute.

3.1.2 Combustion Use Case

Throughout this work we demonstrate our approach applied to a combustion use case, using S3D [16], a direct numerical simulation (DNS) of combustion in turbulence. The combustion simulations in our use case pertain to a class of internal combustion (IC) engine concept, called premixed-charge compression ignition (PCCI). The central idea is that the air-fuel mixture is allowed to ignite on its own, as opposed to being forced to ignite through a spark, as is done in conventional spark-ignited (SI) engines. This results in substantial improvements in fuel efficiency. However, one of the roadblocks to this technology is that the ignition is difficult to control. In particular, it is difficult to predict the precise moment of ignition, which is important for such an engine to be practical. It is undesirable to have simultaneous ignition of the entire mixture, or even a large fraction of the mixture, which would result in knocking, damaging the engine.

Within the broad framework of PCCI, several ideas have been proposed to alleviate the difficulty noted above. All of them seek to control the ignition process by staggering it in time, i.e. different parcels of the air-fuel mixture ignite at different times, so that the overall heat release is delocalized in time. This is done typically by stratifying the mixture, i.e. mixture properties are varied spatially in such a way that the desired heat release profile is obtained. Here, we are interested in two specific techniques called homogeneous-charge compression ignition (HCCI) [7] and reactivity-controlled compression ignition (RCCI) [28, 8]. In both cases, heat release starts out in the form of small kernels at arbitrary locations in the simulation domain, see Figure 3.1. Eventually, multiple kernels ignite as the overall heat release reaches a global maximum and subsequently declines. Since these simulations are computationally and storage-intensive, we want to run the simulation at a coarser grid resolution and save data less frequently during the early build-up phase. When the heat release events occur, we want to run the simulation at the finest grid granularity possible, and store the data as frequently as possible. Therefore, it is imperative to be able to predict the start of the heat release event using an indicator and trigger that serve to inform the application to adjust its grid resolution and I/O frequency accordingly, see Figure 3.2.

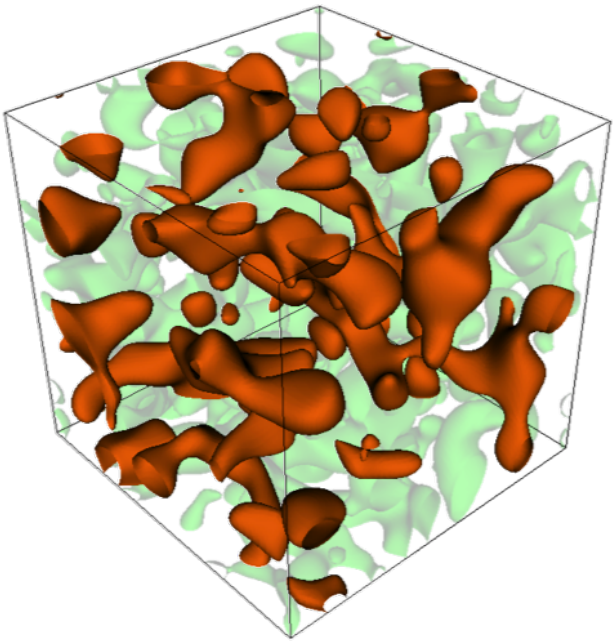


Figure 3.1. In a Homogenous Compression Charge Ignition simulation many small heat kernels slowly develop prior to auto-ignition. In this image, regions of high heat kernels are shown in orange and regions of high vorticity are shown in green.

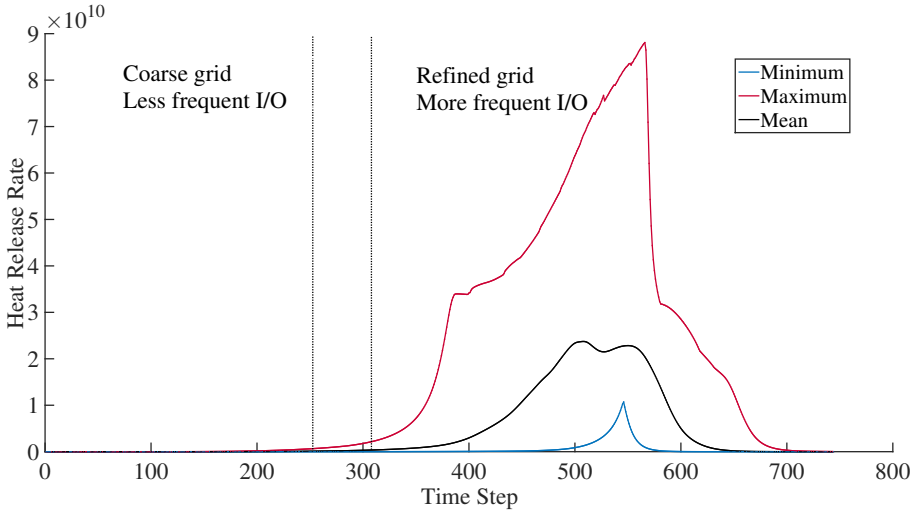


Figure 3.2. The minimum (blue), maximum (red) and mean (black) heat release values for each time step in the simulation. Early in the simulation, we want to run at a coarser grid resolution and save data less frequently. When heat release events occur, we want to run the simulation at the finest granularity, and save the data as frequently as possible. The vertical dotted lines in this figure define a range of time steps within which we would like to make this workflow

3.2 Designing a Noise-Resistant Indicator and Trigger

While general-purpose indicators could be computed (e.g. entropy of a quantity of interest), we argue that application domain-specific indicators in many cases will best capture the phenomena of interest. In this section we describe the design of an indicator and trigger for heat release for our combustion use case. To provide context, we begin by discussing the intuitions that informed our design.

3.2.1 Chemical Explosive Mode Analysis

One of the most reliable techniques to predict incipient heat release is the chemical explosive mode analysis (CEMA). CEMA is a pointwise computational technique described in detail by Lu *et al.* [31] and Shan *et al.* [47]. A brief description is provided here for reference. The conservation equations for reacting species can be written as

$$\frac{D\mathbf{y}}{Dt} = \mathbf{g}(\mathbf{y}) \equiv \omega(\mathbf{y}) + \mathbf{s}(\mathbf{y})$$

The vector $\mathbf{y}$ in CEMA represents temperature and reacting species mass fractions, ω is the reaction source term and $\mathbf{s}$ is the mixing term. The Jacobian of the right hand side can be written as

$$\mathbf{J_g} = \frac{\partial \mathbf{g}(\mathbf{y})}{\partial \mathbf{y}} = \mathbf{J}_\omega + \mathbf{J_s}, \quad where \;\; \mathbf{J}_\omega = \frac{\partial \omega(\mathbf{y})}{\partial \mathbf{y}} \;\; and \;\; \mathbf{J_s} = \frac{\partial \mathbf{s}(\mathbf{y})}{\partial \mathbf{y}}.$$

The chemical Jacobian, $\mathbf{J}_\omega$ can be used to infer chemical properties of the mixture. This is done using an eigen-decomposition of the Jacobian. If the eigenvalues of the Jacobian corresponding to the non-conservative modes are arranged in descending order of the real part, λ_e is defined as the first eigenvalue and λ_i are the remaining eigenvalues. The eigenmode associated with λ_e is defined as a chemical explosive mode (CEM) if

$$\mathrm{Re}(\lambda_e) > 0, \quad for \;\; \lambda_e = \mathbf{b_e} \mathbf{J}_\omega \mathbf{a_e}, \tag{3.1}$$

where $\mathbf{b_e}$ and $\mathbf{a_e}$ are the left and right eigenvectors respectively for λ_e. The presence of a CEM indicates the propensity of a mixture to ignite. CEMA is a pointwise metric, typically computed at every grid point in the simulation domain. The criterion defined above then indicates whether that point will undergo ignition or whether it has already undergone ignition. If it has undergone ignition, we have $Re(\lambda_e) < 0$.

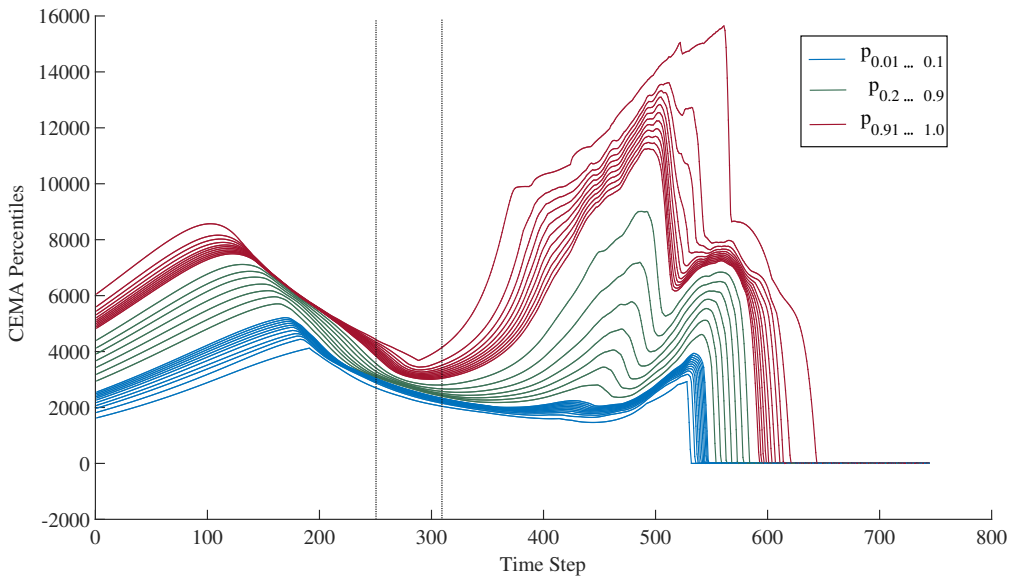


Figure 3.3. In this percentile plot of CEMA values, the lowest blue curve and the highest red curve correspond to the 1 and 100 percentiles ($p_{0.01}$ and p_1), respectively. The blue curves correspond to $p_{0.01\ldots0.1}$, the green curves to $p_{0.2\ldots0.9}$ and the red curves to $p_{0.91\ldots1}$. We notice that as the simulation progresses, the distance between the higher percentiles (red curves) decreases then suddenly increases. Our aim is to define a function that captures when this spread in the high percentiles occurs, as this serves as a good indicator of ignition (it falls within the user-defined window of true trigger time steps, indicated by the vertical dotted lines).

Our CEMA-based indicator is based on global trends of CEMA over time. Consider Figure 3.3 which provides a summary of the trends of CEMA values across all time steps in a simulation. At timestep t, let $\mathcal{C}(t)$ be the array of CEMA values on the underlying mesh. It is convenient to think of $\mathcal{C}(t) \in \mathbb{R}^N$, where N is the total number of grid points. This array is distributed among M processors such that each process accesses N/M points of the field. Let $\widehat{\mathcal{C}(t)}$ be the sorted version of $\mathcal{C}(t)$. For $\alpha \in (0, 1]$, the α-percentile is the entry $\widehat{\mathcal{C}(t)}_{\lceil \alpha N \rceil}$. More specifically, it is the value in $\widehat{\mathcal{C}(t)}$ that is greater than at least $\lceil \alpha N \rceil$ values in $\widehat{\mathcal{C}(t)}$. We denote this value by $p_\alpha(t)$.

We notice that as the simulation progresses, the distance between the higher percentiles (red curves) decreases then suddenly increases. This is illustrated in the plot by the spread in the red curves that occurs just before the dashed line (indicating ignition). What does this mean? Suppose the range of CEMA values (at time t) is $[x, y]$. If the distribution of CEMA values was uniform in $[x, y]$, then the α-percentile would have value around $x + \alpha(y - x)$. Suppose the distribution was highly non-uniform, with a large fraction of small (close to

x) values. Then, for large α, we expect the α-percentile to be smaller than $x + \alpha(y - x)$. Alternately, for (large) $\alpha < \beta$, in the uniform case, the difference between these percentiles is $(\beta - \alpha)(y - x)$. If many values are small, we expect this difference to be larger. Essentially, the gaps between the high percentiles become larger as more CEMA values move towards the lower end of the range.

Our empirical observation is consistent with the underlying physics. From a physical point of view, this trend in the distribution of CEMA values indicates the formation of the first ignition kernels in the fuel-air mixture. As some of these kernels become fully burnt, their CEMA values become negative. As a parcel of fluid transitions from fully unburnt to partially burnt to fully burnt, its temperature increases monotonically and the CEMA value associated with it reaches a peak value before crossing zero and attaining a negative value indicating a fully burnt state. The CEMA values for several other kernels that are in different stages of ignition, i.e. partially burnt, lie between those for unburnt (large positive) and burnt (negative) mixtures. The large range of CEMA values in partially burnt mixtures explains the range of values seen in the percentile plots as ignition is initiated in the mixture.

3.2.2 Designing a Noise-Resistant CEMA-Based Indicator

We introduce an indicator function, *P-indicator* that quantifies the distribution of the top quantiles of CEMA values over time. The P-indicator measures the ratio of the range of the top percentiles to the full range of CEMA values. Our indicator function relies on the range of CEMA values, which are defined by their minimum and maximum. However, the maximum and the minimum of a distribution, by definition, are outliers, and thus they can change drastically between instances even when the underlying distribution does not change. Hence, we avoid the maximum and minimum and replace them with high and low quantiles of the distribution.

Formally, quantiles are defined as values taken at regular intervals from the inverse of the cumulative distribution function of a random variable. For a given data set, quantiles are used to divide the data into equal sized sets after sorting, and the quantiles are the values on the boundary between consecutive subsets. A special case is dividing into 100 equal groups, when we can refer to quantiles as percentiles. This work focuses on percentiles with numbers in the $[0, 1]$ range (although all techniques presented here can be generalized for any quantiles). For example, the 0.5 percentile will refer to the median of the data set.

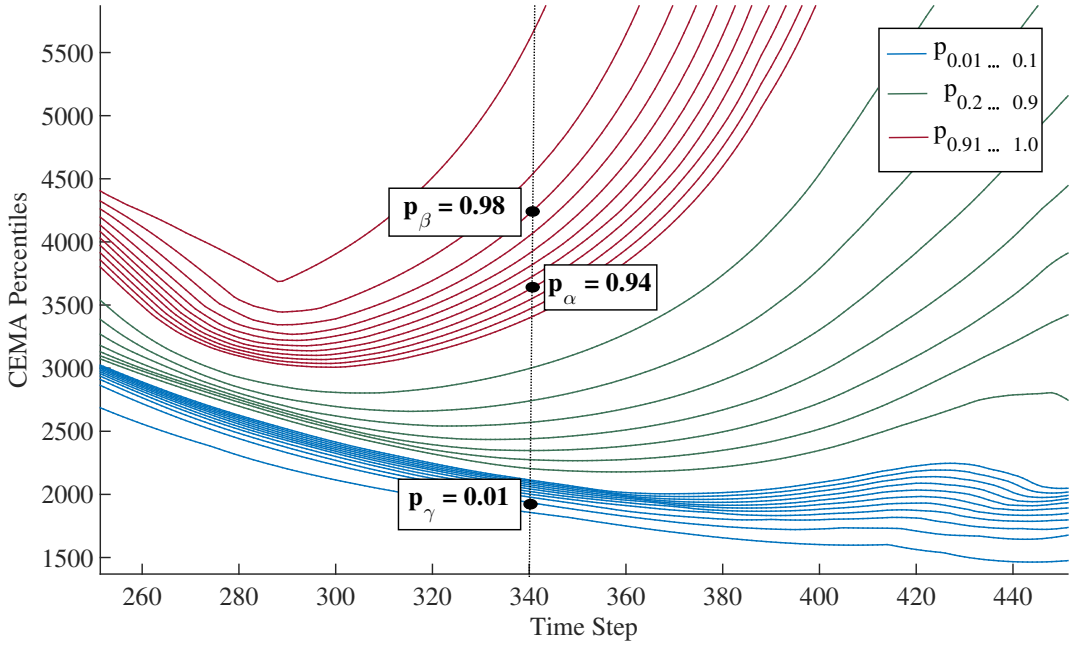


Figure 3.4. $P_{\alpha,\beta,\gamma}(t)$ values evaluated at each time step. Here we illustrate $\alpha_0 = 0.94$, $\beta = 0.98$ and $\gamma = 0.01$ for time step 340 of a simulation.

We substitute the maximum with a high percentile, which we denote by the β-percentile where β is typically in the range $[0.95, 0.99]$ and the minimum with a low percentile, which we denote by the γ-percentile where γ is typically in the range $[0.01, 0.05]$. This substitution provides stability to our measurements, without compromising what we want to measure.

Consider the following notation: Let $A \in \mathbb{R}^N$, be an array in sorted order. The α percentile of this sorted array is exactly the entry $A_{\lceil \alpha N \rceil}$. We use p_α to refer to this value (i.e., $p_\alpha = A_{\lceil \alpha N \rceil}$. The A array will change at each step of the simulation, and thus we will use $A(t)$ and $p_\alpha(t)$ to refer to the data on step t of the simulation.

We define our indicator on a given array A using 3 parameters: α, β and γ. As described above, we use $p_\beta(t)$ and $p_\gamma(t)$ as substitutes for the maximum and the minimum of $A(t)$ respectively, see Figure 3.4. We want to detect whether the range covered by top quantiles shrinks, and α represents the lower end of the top quantiles. Therefore, the range of top quantiles we measure is $[p_\alpha(t), p_\beta(t)]$. In our indicator, we choose $\alpha < \beta$ (typically in the range $[0.95, 0.99]$) and γ (typically in the range $[0.01, 0.05]$). We measure the spread at time t by the P-indicator:

$$P_{\alpha,\beta,\gamma}(t) = \frac{p_\alpha(t) - p_\gamma(t)}{p_\beta(t) - p_\gamma(t)}. \tag{3.2}$$

In this indicator, the denominator corresponds to the full range of CEMA values, while the numerator corresponds to the range after the top quantiles are removed. When the CEMA

values are uniformly distributed, $P_{\alpha,\beta,\gamma}(t) \approx (\alpha-\gamma)/(\beta-\gamma) \approx \alpha/\beta$. When there is a significant shift towards lower values, $P_{\alpha,\beta,\gamma}(t)$ will become smaller.

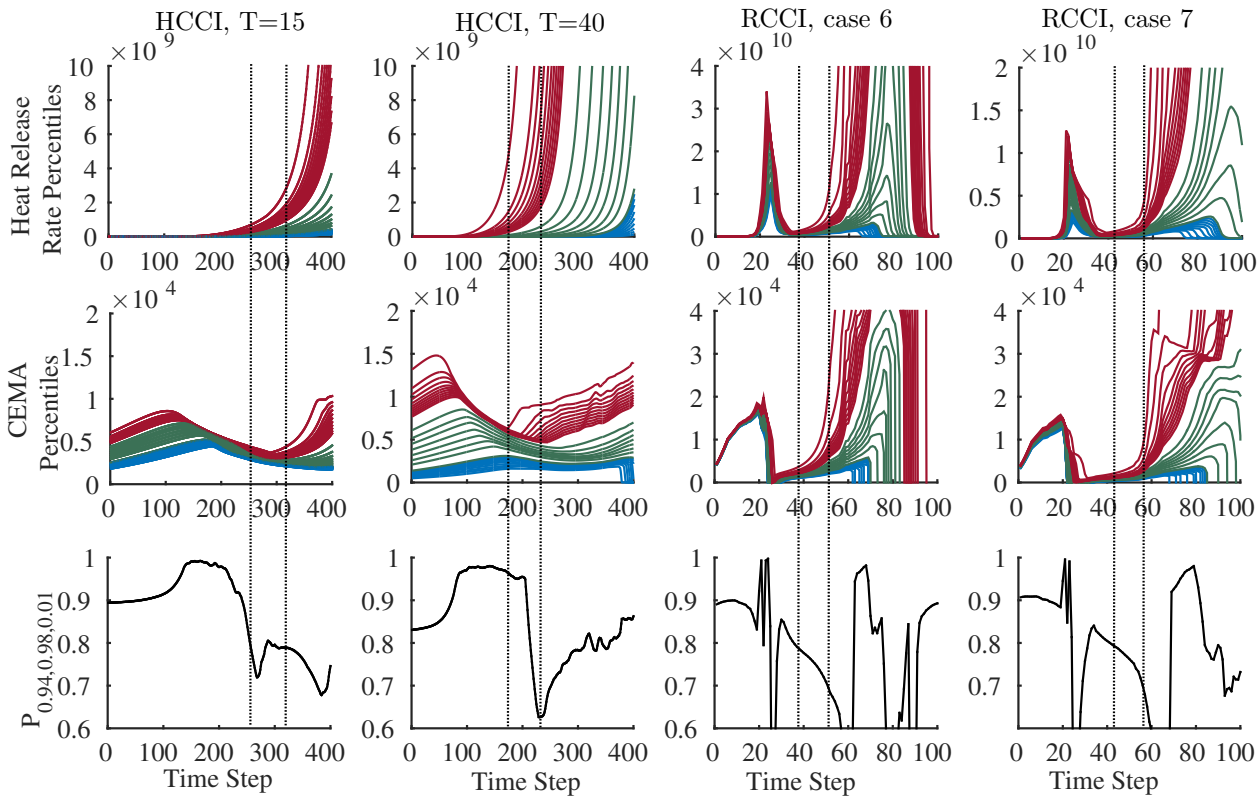


Figure 3.5. Plots showing the (top row) percentiles of the heat release rate, (middle row) percentiles of CEMA, and (bottom row) the P-indicator, as indicated. In the percentile plots, the lowest blue curve and the highest red curve correspond to the 1 and 100 percentiles ($p_{0.01}$ and p_1), respectively. The blue curves correspond to $p_{0.01\ldots0.1}$, the green curves to $p_{0.2\ldots0.9}$ and the red curves to $p_{0.91\ldots1}$. The P-indicator shown is evaluated for $\alpha_0 = 0.94$, $\beta = 0.98$ and $\gamma = 0.01$. The vertical dotted lines crossing the images indicate a window of acceptable "true" trigger time steps, as identified by a domain expert. For the RCCI cases, the trigger time ranges are based on the High Temperature Heat Release (HTHR), i.e., the second peak in the Heat Release Rate (HRR) profiles.

Figure 3.5 illustrates percentile plots for heat release (top row) and CEMA (middle row). In the percentile plots, the lowest blue curve and the highest red curve correspond to the 1 and 100 percentiles ($p_{0.01}$ and p_1), respectively. The blue curves correspond to $p_{0.01\ldots0.1}$, the green curves to $p_{0.2\ldots0.9}$ and the red curves to $p_{0.91\ldots1}$. The P-indicator evaluated using (3.2)

for $\alpha = 0.94$, $\beta = 0.98$ and $\gamma = 0.01$ is shown in the bottom row of Figure 3.5. Results are generated for four test cases described in Table 3.1. The vertical dotted lines were identified by a domain expert who, via examination of heat release and CEMA percentile plots, visually located the time steps in the simulation where the mesh resolution and I/O frequency should be increased. We refer to these time steps as the "true" trigger time steps we wish to identify with the P-indicator and trigger functions. Note, for the RCCI cases, there are two ignition ranges. To simplify the following exposition, we focus on the second rise in the heat release rate profiles, as this is the ignition stage of interest to the scientists. However, we note that our approach is robust in identifying the first ignition stage as well.

Table 3.1. Four Combustion Use Cases analyzed in this study. The "true" trigger time ranges are estimated based on $95 - 100^{\text{th}}$ percentiles of the heat release rate. The computed time ranges were evaluated using our quantile sampling approach. For the RCCI cases, the trigger time ranges are based on the High Temperature Heat Release, i.e., the second peak in the Heat Release Rate profiles.

Problem Instance	Number of Grid Points	Number of Species	"True" Trigger Time Range	Computed Trigger Time
HCCI, T=15	451,584	28	250-315	250-262
HCCI, T=40	451,584	28	175-225	213-220
RCCI, case 6	2,560K	116	38-50	28-45
RCCI, case 7	2,560K–10,240K	116	42-58	35-50

3.2.3 Defining a Trigger

In addition to defining a noise-resistant indicator function, we also need to define a trigger function that returns a boolean value, capturing whether a property of the indicator has been met. Looking at Figure 3.5, we notice that across all experiments from Table 3.1, the P-indicator is decreasing during the true trigger time step windows. Therefore, we seek to find a value $\tau_P \in (0, 1)$, such that $P_{\alpha,\beta,\gamma}(t)$ crosses τ_P *from above*, as the simulation time t progresses.

Figure 3.6 plots the trigger time steps as a function of τ_P for a variety of configurations of α and β for the four use cases desribed in Table 3.1. The horizontal dashed lines indicate the true trigger range identified by our domain expert. We consider those values of τ_P that fall within the horizontal dashed lines to be viable τ_P values for our trigger. We find that across all use cases, there are similar viable ranges of values for τ_P where the predicted trigger time steps do not exhibit large variations. In Figure 3.7 we provide a plot the trigger time steps as a function of τ_P for $P_{\alpha=0.94,\beta=0.98,\gamma=0.01}(t)$ that shows the viable range of τ_P is $[0.725, 0.885]$.

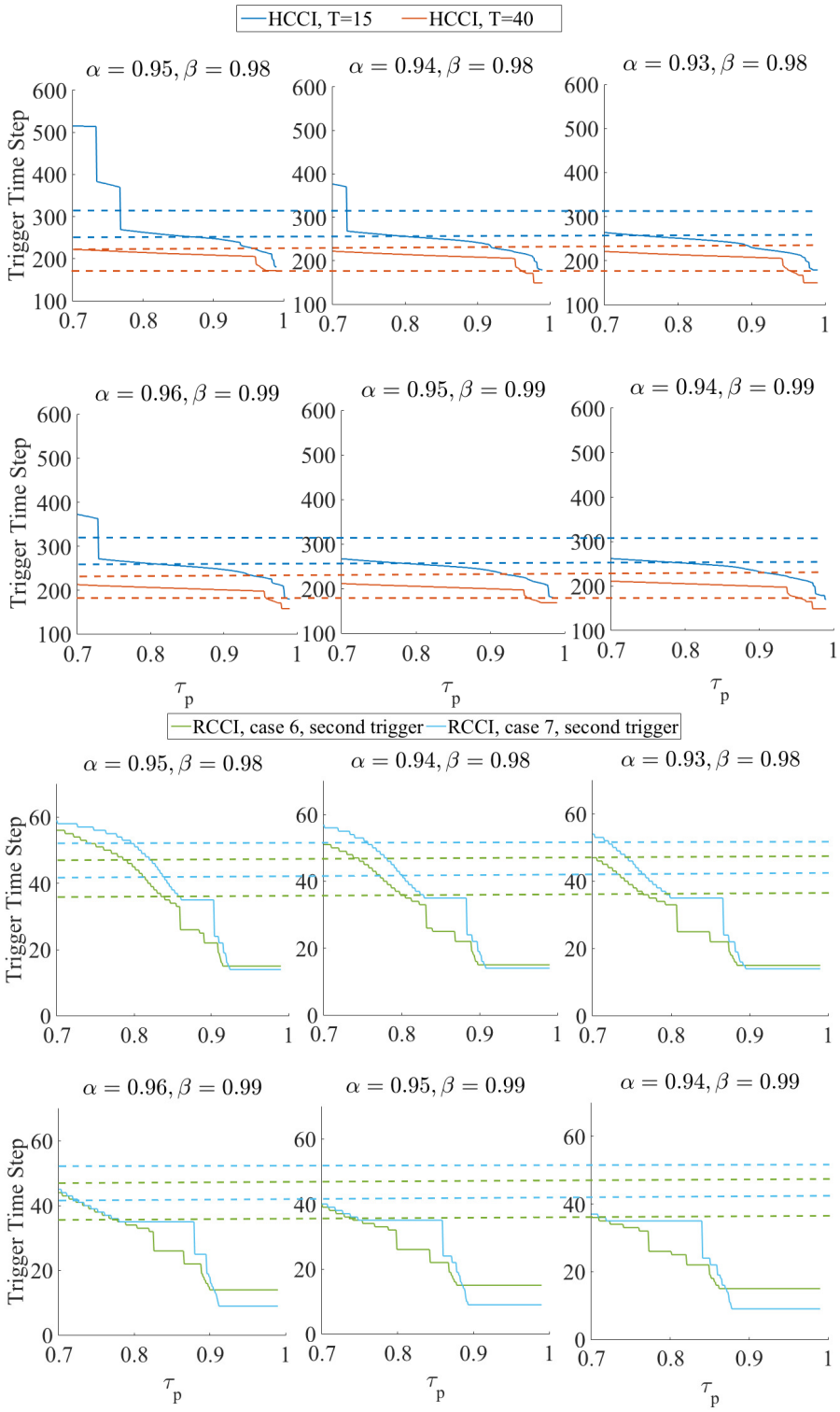


Figure 3.6. This figure shows the trigger time steps as a function of τ_P for a variety of configurations of α and β for the four use cases desribed in Table 3.1. The horizontal dashed lines indicate the true trigger range identified by our domain expert. The P-indicator is evaluated with percentiles computed using all N grid points for the different use cases, as indicated.

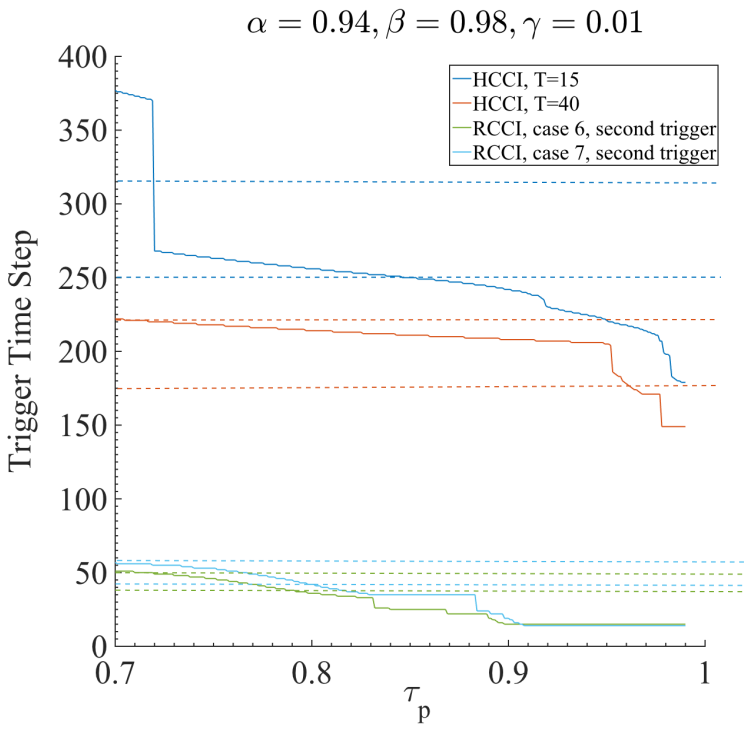


Figure 3.7. This figure plots the trigger time steps as a function of τ_P for $P_{\alpha=0.94,\beta=0.98,\gamma=0.01}(t)$. There is a range of viable values of $\tau_P \in [0.725, 0.885]$ that predict early stage heat release.

3.3 Computing Indicators and Triggers Efficiently: A Sublinear Approach

The previous section showed that a CEMA-based P-indicator and trigger are robust to noise fluctuations and act as a precursor to rapid heat release in combustion simulations. In this section we provide some details regarding its computational cost, which can be prohibitive for large-scale simulations. We then introduce an efficient method to estimate the P-indicator. Our method is based on quantile sampling and it comes with provable bounds on accuracy as a function of the number of samples. Most importantly, the required number of samples for a specified accuracy is independent of the size of the problem, hence our sampling based algorithms offers excellent scalability.

3.3.1 Computational Cost of CEMA

Although CEMA is useful for predicting ignition, it is expensive to compute and thus historically has not been used as a predictive measure. Computing CEMA values involves

constructing a large, dense matrix at every grid point, and computing its eigendecomposition. For the use cases considered here, the time taken to compute the CEMA values scales as the time taken to compute the eigendecomposition of an $M \times M$ matrix, where M is the number of species. Since the Jacobian does not have any spatial structure, the time taken for the CEMA computation step is $\mathbf{O}(M^3)$, which makes it increasingly expensive for larger chemical mechanisms. As seen in Table 3.2, for the ethanol HCCI case presented here that was simulated with 28-species, the cost of computing the CEMA value at every grid point was approximately 5 times the cost of one simulation time-step. The RCCI case, on the other hand included 116-species and the cost was roughly 60 times that of a single time step. Although it is infeasible to compute CEMA at every grid point, our indicator function is defined in terms of distribution of CEMA values, $\mathcal{C}(t)$, and this can be easily approximated by a sampling mechanism, that has provable guarantees on the error.

3.3.2 Approximating Percentiles by Sampling

To remind our notation, we have an array $A \in \mathbb{R}^N$, in sorted order. Our aim is to estimate the α-percentile of A. (We use p_α to denote the percentiles.) Note that this is exactly the entry $A_{\lceil \alpha N \rceil}$. Here is a simple sampling procedure.

1. Sample k independent, uniform indices $r_1, r_2, \ldots, r_k$ in $\{1, 2, \ldots, N\}$.
   Denote by $\widehat{A}$ the sorted array $[A(r_1), A(r_2), \ldots, A(r_k)]$.
2. Output the α-percentile of $\widehat{A}$ as the estimate, $\widehat{p}_\alpha$.

In the next section, we quantitatively show that our estimation, $\widehat{p}_\alpha$, is approximately close to the the true p_α. Such sampling arguments were also used in automatic generation of colormaps for massive data [50].

3.3.3 Theoretical Bounds on Performance

Our analysis relies on the following fundamental result by Hoeffding, which provides a concentration inequality for sums of independent random variables.

Theorem 3.3.1 (Hoeffding [22] or Theorem 1.1 in [17]) *Let $X_1, X_2, \ldots, X_k$ be independent random variables with $0 \leq X_i \leq 1$ for all $i = 1, \ldots, k$. Define $X = \frac{1}{k}\sum_{i=1}^{k} X_i$. For any positive t, we have*

$$\Pr[|X - \mathbf{E}[X]| \geq t] \leq 2\exp(-2t^2/k).$$

Lemma 3.3.2 *Fix α and parameters $\delta, \varepsilon \in (0,1)$, such that $\alpha > \varepsilon$ and $\alpha + \varepsilon \leq 1$. Set $k = \lceil \frac{\log(4/\delta)}{2\varepsilon^2} \rceil$. Then $\widehat{p}_\alpha \in [p_{\alpha-\varepsilon}, p_{\alpha+\varepsilon}]$ with probability at least $1 - \delta$.*

Table 3.2. Additional cost associated with computing CEMA indices at every grid point (no sub-sampling) for two chemical mechanisms. Cost is given in seconds of wall-clock time per overall simulation time step.

Fuel	Mechanism size (species)	Cost without CEMA (sec/ts)	Cost with CEMA (sec/ts)	Cost factor
Ethanol	28	0.3	1.5	5
Primary Reference Fuel (PRF)	116	3.0	180.0	60

Proof: Let X_i be the (Bernoulli) indicator random variable for the event $r_i < (\alpha - \varepsilon)N$. (i.e.,$X_i = 1$, if $r_i < (\alpha - \varepsilon)N$, and zero otherwise.) Note that

$$\mathbf{E}[X_i] = \Pr[X_i = 1] = \Pr[r_i < (\alpha - \varepsilon)N] < \alpha - \varepsilon.$$

By linearity of expectation, $\mathbf{E}[\sum_{i \le k} X_i] < k(\alpha - \varepsilon)$. By Hoeffding's inequality (Theorem 3.3.1),

$$\Pr[\sum_{i \le k} X_i - \mathbf{E}[\sum_{i \le k} X_i] \ge \varepsilon k] \le 2\exp(-2\varepsilon^2 k).$$

The latter is at most $\delta/2$. Thus, with probability $> 1 - \delta/2$,

$$\sum_{i \le k} X_i \le \mathbf{E}[\sum_{i \le k} X_i] + \varepsilon k < \alpha k.$$

In plain English, with probability at least $1 - \delta/2$, the number of random indices strictly less than $(\alpha - \varepsilon)N$ is strictly less than αk.

We repeat a similar argument with indicator random variable Y_i for the event $r_i < (\alpha + \varepsilon)N$. So $\mathbf{E}[Y_i] > \alpha + \varepsilon$ and $\mathbf{E}[\sum_{i \le k} Y_i] > k(\alpha + \varepsilon)$. By Hoeffding's inequality,

$$\Pr[\mathbf{E}[\sum_{i \le k} Y_i] - \sum_{i \le k} Y_i > \varepsilon k] < \delta/2.$$

With probability at least $1 - \delta/2$, the number of random indices at most $(\alpha + \varepsilon)N$ is strictly more than αk.

By the union bound on probabilities, both events hold simultaneously with probability $> 1 - \delta$. In this situation, the α-percentile of the sample lies between the $p_{\alpha-\varepsilon}$ and $p_{\alpha+\varepsilon}$.

It bears emphasizing that the number of samples, k, is independent of the problem size, N, and only depends on ε, δ. So, the required number of samples only depends on the desired accuracy, not on the size of the data. This is the key to the scalability of our approach.

To compute the P-indicator, $P_{\alpha,\beta,\gamma}$, we just employ the procedure above to get estimates $\widehat{p}_\alpha, \widehat{p}_\beta, \widehat{p}_\gamma$. We can use the same samples (with only an additive increase to k) for all estimates,

so we do not have to repeat the procedure 3 times. That yields the approximate P-indicator, denoted by $\widehat{P}_{\alpha,\beta,\gamma}$.

Theorem 3.3.3 *Fix α, β, γ and parameters $\delta, \varepsilon \in (0,1)$ such that $\alpha < \beta - 2\varepsilon$ and $\varepsilon < \min(\alpha, \beta, \gamma)$. Set $k = \lceil \frac{\log(12/\delta)}{2\varepsilon^2} \rceil$. With probability $> 1 - \delta$,*

$$\widehat{P}_{\alpha,\beta,\gamma} \in [P_{\alpha-\varepsilon,\beta+\varepsilon,\gamma+\varepsilon}, P_{\alpha+\varepsilon,\beta-\varepsilon,\gamma-\varepsilon}].$$

Proof: Apply Lem. 3.3.2 for each of α, β, γ with error parameter $\delta/3$. That gives the value of k as stated in the theorem. By the union bound on probabilities, all the following hold simultaneously with probability $> 1 - \delta$: $\widehat{p}_\alpha \in [p_{\alpha-\varepsilon}, p_{\alpha+\varepsilon}]$, $\widehat{p}_\beta \in [p_{\beta-\varepsilon}, p_{\beta+\varepsilon}]$, and $\widehat{p}_\gamma \in [p_{\gamma-\varepsilon}, p_{\gamma+\varepsilon}]$, Hence,

$$\widehat{P}_{\alpha,\beta,\gamma} = \frac{\widehat{p}_\alpha - \widehat{p}_\gamma}{\widehat{p}_\beta - \widehat{p}_\gamma} \leq \frac{p_{\alpha+\varepsilon} - \widehat{p}_\gamma}{p_{\beta-\varepsilon} - \widehat{p}_\gamma} \leq \frac{p_{\alpha+\varepsilon} - p_{\gamma-\varepsilon}}{p_{\beta-\varepsilon} - p_{\gamma-\varepsilon}}$$

For the last inequality, observe that for fixed $x < y$, $(x - z)/(y - z)$ is a decreasing function of z. An analogous argument proves the lower bound for $\widehat{P}_{\alpha,\beta,\gamma}$.

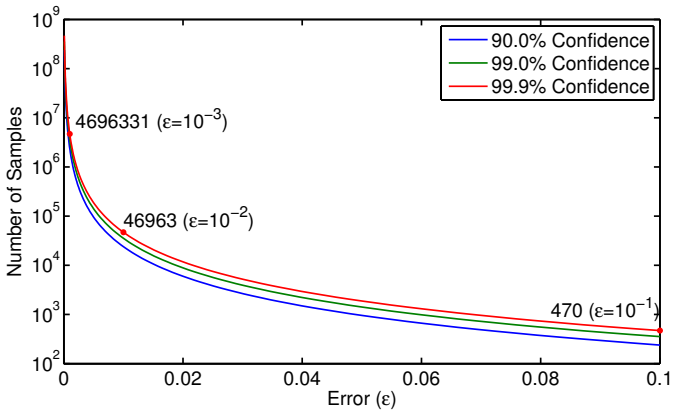


Figure 3.8. The number of samples needed for different error rates and different levels of confidence. A few data points at 99.9% confidence are highlighted.

Figure 3.8 shows the number of samples needed for different error and confidence rates. We show three different curves for difference confidence levels. Increasing the confidence has minimal impact on the number of samples. The number of samples is fairly low for error rates of 0.1 or 0.01, but it increases with the inverse square of the desired error. Nonetheless, the four million samples required for an error rate of $\varepsilon = .001$ at 99.9% confidence requires only tens of samples per processor at the extreme scales. In practice, $\varepsilon = .01$ at 99.9% confidence, thus 48,000 samples was enough to compute robust estimates.

Interpretation of the Bounds

Our bounds on quantile estimations are based on which quantile we sample. That is our sampling algorithm may return the $\alpha + \varepsilon$-th quantile instead of α-th quantile, and we can quantify this error, ε, as a function of the number of samples. However, it does not quantify the difference between p_α and $\widehat{p}_\alpha = p_{\alpha+\varepsilon}$. Subsequently, Theorem 3.3.3 shows that the range we sample can be made arbitrarily close to the original range by increasing the number of samples, and bounds the error in the range for any given sample size. Yet, it does not bound the difference between $P_{\alpha,\beta,\gamma}$ and $\widehat{P}_{\alpha,\beta,\gamma}$, which depends on the distribution of the data. However, this is not a critical issue for our purposes, as we argue next, and empirically verify with experimental results in the next section.

The difference between $P_{\alpha,\beta,\gamma}$ and $\widehat{P}_{\alpha,\beta,\gamma}$ will be disproportional to ε only when there are gaps in the distribution around one of the three parameters, α, β, or γ. Note that these three parameters are user specified, and they are used to quantify the range of top percentiles. If the underlying distribution is such that we expect many such gaps frequently, then our metric itself will be extremely sensitive to the choice of the input parameters, even if compute the metric exactly. That is our metric should not produce vastly different results when we choose $\alpha = 0.940$ or $\alpha = 0.941$. However, there is still a possibility that such gaps may form, just like any other low probability event. One trick to improve robustness of our metric against such low probability events is to pick the input parameters randomly from a specified range. That is instead of specifying γ as 0.03 we can pick it randomly in the range $[0.02, 0.04]$. By such randomness, even if there is a gap at point, ϕ, the probability that ϕ is in the $[\beta - \varepsilon, \beta + \varepsilon]$ range will approach to zero with increasing number of samples and thus decreasing ε.

However, we want to note that this is only a theoretical exercise. From a practical perspective, we have not observed any gaps in the distribution and $p_{\alpha+\varepsilon}$ is a good estimate for p_α, and it gets better with more samples. For the experiments in the remainder of this work, we have not used the randomization technique when choosing the parameters, γ, α, and β.

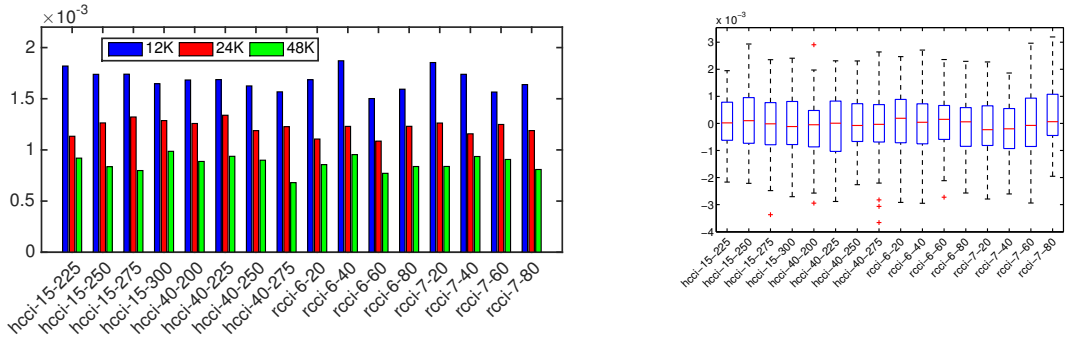
3.3.4 Empirical Evaluation of the Sampling Based Algorithm

In this section we present our empirical evaluation of the proposed sampling techniques. Experiments in this section will focus on only the evaluation of the proposed algorithm, since we first want to verify that the proposed sampling technique accurately estimates the P-indicator. In the next section, we will put all pieces together and study how the P-indicator and the proposed technique perform together in-situ as the simulation is running.

In the first set of experiments, we investigate the error in quantile ranges. For these experiments, we use 16 randomly selected instances of CEMA distributions from various HCCI and RCCI simulations. These instances are named such that the first part refers to the simulation type and case, and the last part refers to the time step. We use sampling

to estimate the $\alpha = 0.94$ percentile, which returns an entry from the distribution. Then we check the percentile of this entry in the full data, say $\alpha + \varepsilon$. In the first set of experiments, we focus on this difference ε, which we bounded in our theoretical analysis.

Figure 3.9 (a) presents the results of our investigation into the error of quantile ranges for various data sets and increasing number of samples: 12,000, 24,000, and 48,000. For this figure, we ran our sampling algorithm 100 times for each instance (i.e., a data set and number of samples combination) and computed ε. The figure presents the average $|\varepsilon|$ for each instance. As the figure shows, sampling yields accurate estimations in all data sets, and the error drops with increasing number of samples. It becomes extremely small for 48,000 samples. Here an error of 0.001 means we will be using $p_{0.941}$ quantile instead of $p_{0.940}$. We did not find it necessary to investigate increasing the number of samples further.



(a)Average error

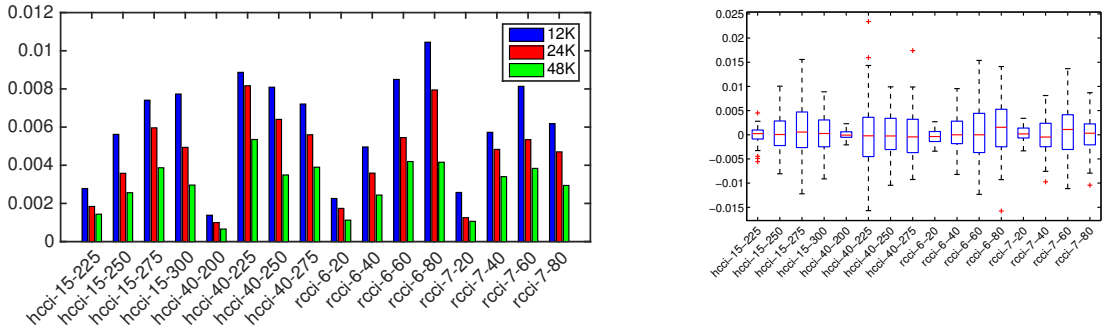
(b) Distribution of errors

Figure 3.9. (a) Error in the percentile sampled as the average of absolute values of 100 runs on various data sets with increasing number of samples. (b) The distribution of errors of each data set for the percentile study using 48,000 samples.

In Figure 3.9 (b), we show the distribution of errors in percentiles sampled for the 100 runs of each dataset using 48,000 samples. In this plot, the central mark (red) shows the median error, while the edges of the (blue) box are the 25th and 75th percentiles. The whiskers extend to the most extreme points considered not to be outliers, and the outliers (red plus marks) are plotted individually. As this figure shows, the estimates are consistently accurate, and the results in practice are much better than those indicated by Theorem 3.3.3. According this theorem, 48,000 samples lead to an error of ≈ 0.01 with a confidence of %99. This means in 100 experiments we expect to have 1 run for which the error is > 0.01. However, in the 16 data sets with 100 runs each the maximum error was 0.005, half of what the upper bound indicates. These results show that sampling enables us to sample a quantile that is consistently accurate.

In the next set of experiments, we look at how our indicator is affected by the minor errors in the quantile. More specifically, we want to see how the difference between p_α and $p_{\alpha+\varepsilon}$ affect our indicator. For those experiments, we used 16 randomly selected instances

of CEMA distributions from various HCCI and RCCI simulations and computed the P-indicator exactly, and using sampling with parameters, $\gamma = 0.01$ $\alpha = 0.94$, and $\beta = 0.98$. While we repeated the same experiments with different parameters, we have not observed



(a) Average Error

(b) Distribution of errors

Figure 3.10. (a) Errors in estimation of the P-indicator for increasing number of samples. (b) The distribution of errors in estimation of the P-indicator for 48,000 samples.

any sensitivities of our tests to the choice the parameters, hence we will present results for only this setting.

Figure 3.10(a) presents results for our indicator tests for various data sets and increasing number of samples: 12,000, 24,000, and 48,000. For this figure, we ran our sampling algorithm 100 times on each data set and number of samples, and computed the difference between exact and estimated values of the P-indicator for each instance. The figure presents the average absolute error for each instance. As the figure shows, sampling produces accurate estimations in all data sets, and the error drops with increasing number of samples. The bounds on Theorem 3.3.3 does not apply in this case. but regardless, the errors are very small, and certainly sufficient to detect any trend in the distribution of the underlying values.

Figure 3.10(b) shows the distribution of errors for the P-indicator test for 100 runs of each dataset using 48,000 samples. In this plot, the central mark (red) shows the median error, while the edges of the (blue) box are the 25th and 75th percentiles. The whiskers extend to the most extreme points considered not to be outliers, and the outliers (red plus marks) are plotted individually. As this figure shows, the estimates are consistently accurate.

Lastly, we performed a series of experiments examining the variation in the trigger time steps as a function of the number of samples used per processor. The data for Figure 3.11 was generated via 200 realizations of the P-indicator with $\alpha = 0.94$, $\beta = 0.98$ and $\gamma = 0.01$, and τ_P drawn from $[0.725, 0.885]$. The horizontal dashed lines in this figure define the range of true trigger time steps within which we would like to make the workflow transition (as identified by a domain expert). This plot demonstrates that, even across a wide range of τ_P values, with a small number of samples per processor, the quantile sampling approach can

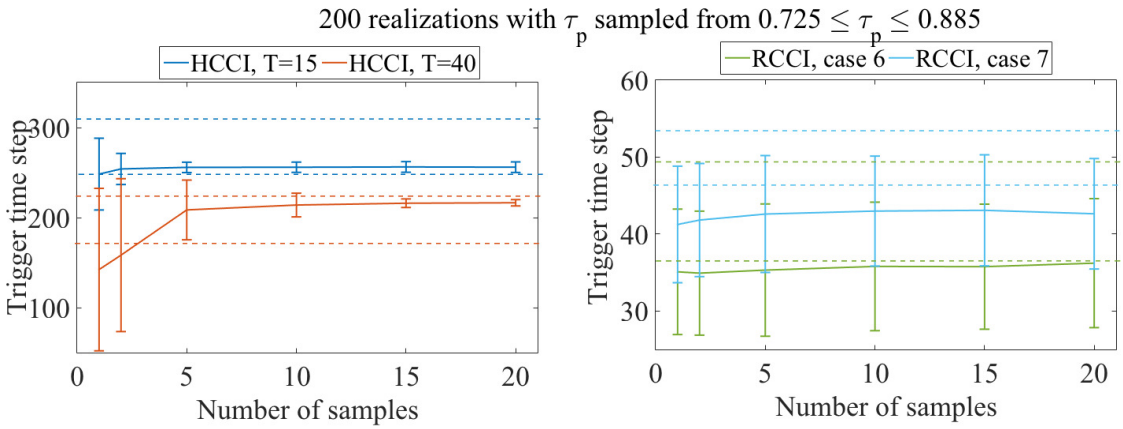


Figure 3.11. Plots illustrating the variability of the trigger time steps predicted by the P-indicator and trigger as a function of the number of samples per processors. The data for these plots was generated via 200 realizations of the P-indicator with $\alpha = 0.94$, $\beta = 0.98$ and $\gamma = 0.01$, and with τ_P drawn from $[0.725, 0.885]$. The horizontal dashed lines define the range of time steps within which we would like to make the workflow transition (as identified by a domain expert).

accurately estimate the true trigger time steps as defined by the domain expert. The next step will be putting all the pieces together to see how we can predict heat release in-situ, within a simulation run.

3.4 Putting All Pieces Together: Diagnosing Heat Release With Sublinear Algorithms in S3D

The algorithm described in the previous section was deployed in-situ for a two-dimensional direct numerical simulation (DNS) of the ethanol HCCI problem (Case HCCI, T=40 in Table 3.1). The DNS was run on half a million grid points with 784 processors. 20 points were sampled at random from each processor for the CEMA analysis, generating a total of 15680 samples for computing the trigger with the P-indicator. The parameters for computing the metric were chosen as follows: $\alpha = 0.94$, $\beta = 0.98$ and $\gamma = 0.01$. This corresponds to less than 4% of the total simulation volume.

Figure 3.12 shows the P-indicator being computed in-situ inside the simulation code. From top to bottom, the rows show the result when the indicator is computed every 10, 100 and 1000 time steps. As the frequency of computing these indicator increases, the signal tends to get noisier. However, the overall trends and triggers do not change. These images show that our quantile-sampling approach provides a well defined trigger, using $\tau_P \in [0.725, 0.885]$ and can be used with confidence to predict the rapid rise in the heat release rate that we require to guide temporal and spatial refinement decisions in-situ.

As can be inferred from Table 3.2, performing the CEMA analysis on all grid points would increase the cost of the simulation by a factor of 5, or 400%, which is clearly infeasible. Using the sublinear sampling algorithm on the other hand, incurs an overhead of only 1% on the total simulation cost when performed every ten time-steps. The cost savings are even more dramatic in larger, three-dimensional production runs, as the number of samples required does not increase with the number of grid points. Furthermore, we note the cost savings are further increased for larger mechanisms such a primary reference fuel (PRF), composed of a blend of iso-octane and n-heptane. For the PRF mechanism, the CEMA overhead without sublinear sampling would be a factor of 60 or more. We plan to deploy this algorithm in-situ in future large three-dimensional production simulations using S3D, especially with large chemical mechanisms such as PRF.

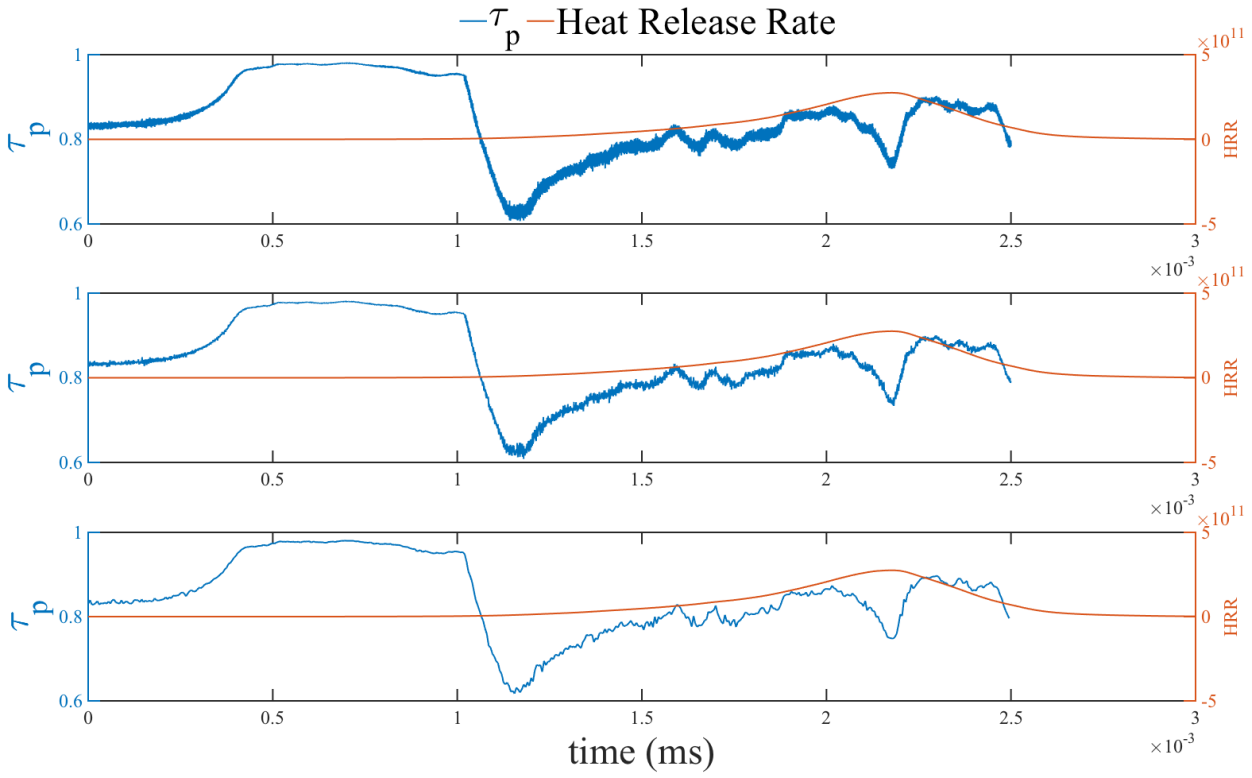


Figure 3.12. Plots showing P-indicator being computed every 10 (top), 100 (middle) and 1000 (bottom) time steps.

Chapter 4

Accomplishments

This Chapter highlights the accomplishments and outputs of this projects in terms of publications, presentations, and follow-on funding that was obtained as a result of this research.

Publications

- *Data-driven visual exploration of extreme-scale time varying data.* D. Thompson, D. Rogers, N. Fabian, J. Wendelberger, F. Samsel, T. Turton, J. Bennett, A. Pinar (in preparation)
- *A dashboard for comparative visualization of many-variate data.* D. Thompson, P. Pebay, H. Kolla, J. C. Bennett, A. Pinar (in preparation)
- *Trigger detection for adaptive scientific workflows using percentile sampling.* J. Bennett, A. Bhagatwala, J. Chen, C. Seshadhri, A. Pinar, M. Salloum, to appear in: SIAM Journal on Scientific Computing. (SAND2015-5112)
- *Enabling adaptive scientific workflows via trigger detection.* M. Salloum, J. Bennett, A. Pinar, A. Bhagatwala, J. Chen, to appear in In Situ Infrastructures for Enabling Extreme-scale Analysis and Visualization 2015 (SAND2015-6992 C)
- *Sublinear Algorithms for Extreme-scale Data Analysis.* C. Seshadhri, A. Pinar, D. Thompson (Kitware), and J. Bennett, in Topological and Statistical Methods for Complex Data: Tackling Large-Scale, High-Dimensional, and Multivariate Data Sets Springer, Nov 2014. (SAND2014-16025 B)
- *A provably-robust sampling method for generating colormaps of large data.* D. Thompson, J. Bennett, C. Seshadhri, and A. Pinar, in Proc. IEEE Symposium on Large-Scale Data Analysis and Visualization (LDAV), 2013. (SAND2013-4171C)
- *Avoiding the Global Sort: A Faster Contour Tree Algorithm.* B. Raichel, C. Seshadhri (under review)

Presentations

- *A project in the life of a data scientist*, ICERM workshop on Mathematics in Data Science, Janine Bennett (Invited)
- *Extreme-Scale In situ Data Analysis* Broader Engagement Panel, Supercomputing 2014, New Orleans, LA. Nov 16, 2014 (invited)
- *Sublinear Algorithms for In-situ & In-transit Data Analysis at Exascale*, DOE ASCR

Exascale Math Working Group Workshop (EXAMath13), Ali Pinar, Washington DC, August, 2013.

- *Challenges in Turning Science Data into Knowledge* Janine Bennett SOS 17, Jekyll Island, GA April 2013 (invited)
- *A provably-robust sampling method for generating colormaps of large data.* (LDAV), 2013.

Follow on Funding In 2014 an New ASCR sponsered project was funded, titled *A Unified Data-Driven Approach for Programming In-Situ Analysis and Visualization.* This is a three year project, with Sandia's effort funded at $405K/year. Other collaborators include Los Alamos National Laboratory, Kitware Inc, the University of Utah, and Stanford. The overarching goal of this project is to explore and develop a data-driven approach for developing in-situ-enabled applications on future extreme-scale, high-performance systems. The success of this approach will be evaluated by exploring advanced data reduction algorithms and key data analysis and visualization operations including sublinear techniques. Some of the research efforts on this upcoming ASCR project are ideas that were generated as a direct result of the accomplishments and research performed by this LDRD.

Chapter 5

Conclusions and Lessons Learned

This LDRD has made several key contributions, highlighting the potential for sublinear algorithms to mitigate HPC challenges posed by next generation computing platforms. In particular, this LDRD research has focused on addressing several fundamental issues posed by the shift to in-situ, concurrent workflows. We have explored how sublinear algorithms could be leveraged to make the analysis algorithms themselves more efficient. Towards this end, we have developed a sampling based method to generate colormaps. Our algorithm identifies important values in the data, provides a provably-good approximation of the CDF of the remaining data, and uses this information to automatically generate discrete and/or continuous color maps. Our experiments showed the new colormaps yield images that better highlight features within the data. The proposed approach is simple and efficient, yet provably robust. Most importantly, the number of samples required by the algorithm depends only on desired accuracy in estimations and is independent of the dataset size. This provides excellent scalability for our algorithms, as the required preprocessing time remains constant despite increasing data as we move to exascale computing. The algorithms are also efficiently parallelizable and and yield linear scaling. These algorithms have been made available to the broader scientific community in Paraview and VTK, both open source Kitware, Inc. products.

We have also developed an approach for enabling dynamic, adaptive, extreme-scale scientific computing workflows. We introduce the notion of indicators and triggers that are computed in-situ, that support data-driven control-flow decisions based on the simulation state. For those indicators and triggers that are computationally prohibitive to compute, we demonstrate how sublinear algorithms enable their estimation with high confidence while incurring extremely low computational overheads.

In the context of our adaptive workflow research, we demonstrate our approach in practice using a proposed indicator to detect changes in the underlying physics of a combustion simulation. The goal of the indicator is to predict rapid heat release in direct numerical simulations of turbulent combustion. We show that chemical explosive mode analysis (CEMA) can be used to devise a noise-tolerant indicator for rapid increase in heat release. Specifically, we study the distribution of CEMA values, and show that heat release is preceded by a decrease in the range of top quantiles in this distribution. We devise an indicator to quantify this intuition and show that it can serve as a robust precursor for heat release. The cost of exhaustive computation of CEMA values dominates the total simulation time, and we

propose a sublinear algorithm based on quantile sampling to overcome this computational bottleneck. Our algorithm comes with provable error/confidence bounds, as a function of the number of samples. Most importantly, the number of samples is independent of the problem size, thus our proposed sampling algorithm offers perfect scalability. Our experiments show that our sublinear algorithm is nearly as accurate as an exact algorithm that relies on exhaustive computation, while taking only a fraction of the time. Essentially, sampling in this case provides the algorithmic foundation to turn a critical yet intractable indicator into a practical indicator that takes negligible time. Our experiments on homogeneous charge compression ignition (HCCI) and reactivity controlled compression ignition (RCCI) simulations show that the proposed method can predict heat release, and its computational overhead is negligible.

One of the risks identified from the start with our project was that of adoption. While it might be possible to generate some very interesting tools that leverage sublinear algorithms, those tools are not of great value if scientists are wary to use them. To mitigate this risk, throughout this LDRD, we worked very closely with application developers, particularly in the area of combustion. Our goal was to develop new theory, demonstrate results in practice on current problems, and demonstrate the potential of our approach to mitigate exascale research challenges. Through our socialization with the scientists, some interesting findings emerged. First, *sampling is not always computationally efficient enough to warrant the tradeoff in accuracy.* In an HPC scientific simulations, the mesh and associated data are distributed across processing elements. Data decomposition is optimized for the simulation and communication costs end up dominating runtime. Overall, we found that *scientists werent interested in sampling of final quantities of interest.* The reasons were two-fold: they wanted to maintain reproducibility of their results, and they wanted teh ability to compare their results with those of previous approaches. However, *scientists were very interested to leverage sampling to enable new capabilities.* This included things like improved visual debugging, the enabling more in-depth post-processing at higher temporal resolution, and the judicious use of resources (I/O, compute time) via adaptive, data-driven workflows. Lastly, we found that *combined empirical and theoretical proofs were critical to adoption.*

In summary, the impact of our LDRD research is two fold. From the applications' perspective, we have introduced an important tool that enables feature detection and adaptive workflows in their simulations, and have demonstrated results in both cases for combustion, an important area of computational science and engineering. Our proposed methods, enable the controlling of identification of features, mesh granularities, and adaptive I/O frequencies. It is already becoming critically important to have adaptive control over these quantities, but will be crucial as we look ahead to exascale computing. From an algorithmic perspective, our efforts showcased how sublinear algorithms, a recent development in theoretical computer science can be applied in-situ. We believe these algorithmic techniques hold great potential for in-situ analysis, and we expect them to be more widely used in the near future.

References

[1] Hasan Abbasi, Greg Eisenhauer, Matthew Wolf, Karsten Schwan, and Scott Klasky. Just In Time: Adding Value to The IO Pipelines of High Performance Applications with JITStaging. In *Proc. of 20th International Symposium on High Performance Distributed Computing (HPDC'11)*, June 2011.

[2] Sean Ahern, Arie Shoshani, Kwan-Liu Ma, Alok Choudhary, Terence Critchlow, Scott Klasky, Valerio Pascucci, Jim Ahrens, E. Wes Bethel, Hank Childs, Jian Huang, Ken Joy, Quincey Koziol, Gerald Lofstead, Jeremy S Meredith, Kenneth Moreland, George Ostrouchov, Michael Papka, Venkatram Vishwanath, Matthew Wolf, Nicholas Wright, and Kensheng Wu. *Scientific Discovery at the Exascale, a Report from the DOE ASCR 2011 Workshop on Exascale Data Management, Analysis, and Visualization*. 2011.

[3] Grey Ballard, Tamara G. Kolda, Ali Pinar, and C. Seshadhri. Diamond sampling for approximate maximum all-pairs dot-product (mad) search. Technical Report Arxiv:1506.03872, 2015. to appear in Proc. ICDM 2015.

[4] G. Bansal. *Computational Studies of Autoignition and Combustion in Low Temperature Combustion Engine Environments*. PhD thesis, University of Michigan, 2009.

[5] J. Bennett, A. Bhagatwala, J. Chen, A. Pinar, M. Salloum, and C. Seshadhri. Trigger detection for adaptive scientific workflows using percentile sampling. Technical Report Arxiv:1506.08258, 2015.

[6] Janine C. Bennett, Hasan Abbasi, Peer-Timo Bremer, Ray Grout, Attila Gyulassy, Tong Jin, Scott Klasky, Hemanth Kolla, Manish Parashar, Valerio Pascucci, Philippe Pebay, David Thompson, Hongfeng Yu, Fan Zhang, and Jacqueline Chen. Combining in-situ and in-transit processing to enable extreme-scale scientific analysis. In Jeffrey Hollingsworth, editor, *SC '12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, Salt Lake Convention Center, Salt Lake City, UT, USA, November 10–16, 2012*, pages 49:1–49:9, pub-IEEE:adr, 2012. IEEE Computer Society Press.

[7] A. Bhagatwala, J. H. Chen, and T. Lu. Direct numerical simulations of SACI/HCCI with ethanol. *Comb. Flame*, 161:1826–1841, 2014.

[8] A. Bhagatwala, R. Sankaran, S. Kokjohn, and J. H. Chen. Numerical investigation of spontaneous flame propagation under RCCI conditions. *Comb. Flame*, Under review.

[9] Udeepta Bordoloi and Han-Wei Shen. View selection for volume rendering. In *Vis '05: Proceedings of the IEEE Visualization 2005*, pages 487–494, 2005.

[10] Michelle A. Borkin, Krzysztof Z. Gajos, Amanda Peters, Dimitrios Mitsouras, Simone Melchionna, Frank J. Rybicki, Charles L. Feldman, and Hanspeter Pfister. Evaluation of artery visualizations for heart disease diagnosis. *IEEE Transactions on Visualization and Computer Graphics*, 17(12), 2011.

[11] David Borland and Russell M. Taylor II. Rainbow color map (still) considered harmful. *IEEE Computer Graphics & Applications*, pages 14–17, March 2007.

[12] Jean-Marie Favre Brad Whitlock and Jeremy S. Meredith. Parallel In Situ Coupling of Simulation with a Fully Featured Visualization System. In *Proc. of 11th Eurographics Symposium on Parallel Graphics and Visualization (EGPGV'11)*, April 2011.

[13] Cynthia A. Brewer. Guidelines for selecting colors for diverging schemes on maps. *The Cartographic Journal*, 33(2):79–86, 1996.

[14] Cynthia A. Brewer. A transition in improving maps: The ColorBrewer example. *Cartography and Geographic Information Science*, 30(2):159–162, 2003.

[15] Cynthia A. Brewer, Geoffrey W. Hatchard, and Mark A. Harrower. ColorBrewer in print: A catalog of color schemes for maps. *Cartography and Geographic Information Science*, 30(1):5–32, 2003.

[16] J. H. Chen, A. Choudhary, B. de Supinski, M. DeVries, E. R. Hawkes, S. Klasky, W. K. Liao, K. L. Ma, J. Mellor-Crummey, N. Podhorski, R. Sankaran, S. Shende, and C. S. Yoo. Terascale direct numerical simulations of turbulent combustion using S3D. *Computational Science and Discovery*, 2:1–31, 2009.

[17] Devdatt Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

[18] Martin Eisemann, Georgia Albuquerque, and Marcus Magnor. Data driven color mapping. In *Proceedings of EuroVA: International Workshop on Visual Analytics*, Bergen, Norway, May 2011.

[19] N. Fabian, K. Moreland, D. Thompson, A.C. Bauer, P. Marion, B. Gevecik, M. Rasquin, and K.E. Jansen. The paraview coprocessing library: A scalable, general purpose in situ visualization library. In *Proc. of IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, pages 89 –96, October 2011.

[20] E. Fischer. The art of uninformed decisions: A primer to property testing. *Bulletin of EATCS*, 75:97–126, 2001.

[21] K. R. Gabriel and J. Neumann. A Markov chain model for daily rainfall occurrence at Tel Aviv. *Quarterly Journal of Royal Meteorology Society*, 88:90–95, 1962.

[22] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

[23] N. T. Ison, A. M. Feyerherm, and L.D. Bark. Wet period precipitation and the gamma distribution. *Journal of Applied Meteorology*, 10:658–665, 1971.

[24] Madhav Jha, C. Seshadhri, and Ali Pinar. Path sampling: A fast and provable method for estimating 4-vertex subgraph counts. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pages 495–505, Republic and Canton of Geneva, Switzerland, 2015. International World Wide Web Conferences Steering Committee.

[25] Madhav Jha, C. Seshadhri, and Ali Pinar. A space-efficient streaming algorithm for estimating transitivity and triangle counts using the birthday paradox. *ACM Trans. Knowl. Discov. Data*, 9(3):15:1–15:21, February 2015.

[26] Gordon Kindlmann, Erik Reinhard, and Sarah Creem. Face-based luminance matching for perceptual colormap generation. In *Proceedings of the IEEE Visualization*, 2002.

[27] Joe Kniss, Gordon Kindlmann, and Charles Hansen. Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):270–285, 2002.

[28] S. L. Kokjohn, R. M. Hanson, D. A. Splitter, and R. D. Reitz. Fuel reactivity controlled compression ignition (rcci): A pathway to controlled high-efficiency clean combustion. *Int. J. Engine. Res.*, 12:209–226, 2011.

[29] Tamara G. Kolda, Ali Pinar, Todd Plantenga, C. Seshadhri, and Christine Task. Counting triangles in massive graphs with mapreduce. *SIAM Journal on Scientific Computing*, 36(5):S48–S77, 2014.

[30] G.W. Larson, H. Rushmeier, and C. Piatko. A visibility matching tone reproduction operator for high dynamic range scenes. *IEEE Transactions on Visualization and Computer Graphics*, 3(4), December 1997.

[31] T. Lu, C. S. Yoo, J. H. Chen, and C. K. Law. Three-dimensional direct numerical simulation of a turbulent lifted hydrogen flame in heated coflow: a chemical explosive mode analysis. *J. Fluid Mech.*, 652:45–64, 2010.

[32] David Lewis MacAdam. "visual sensitivities to color differences in daylight". *JOSA*, 32(5):247–274, 1942.

[33] Marti Maria. Little CMS: How to use the engine in your applications, 2012.

[34] Kenneth Moreland. Diverging color maps for scientific visualization. In George Bebis, Richard D. Boyle, Bahram Parvin, Darko Koracin, Yoshinori Kuno, Junxian Wang, Renato Pajarola, Peter Lindstrom, André Hinkenjann, Miguel L. Encarnação, Cláudio T. Silva, and Daniel S. Coming, editors, *Advances in Visual Computing, 5th International Symposium, ISVC 2009, Las Vegas, NV, USA, November 30 - December 2, 2009, Proceedings, Part II*, volume 5876 of *Lecture Notes in Computer Science*, pages 92–103. Springer, 2009.

[35] K. Myers, E. Lawrence, M. Fugate, J. Woodring, J. Wendelberger, and J. Ahrens. An in situ approach for approximating complex computer simulations and identifying important time steps. Technical report, 2014. arXiv:1409.0909v1.

[36] Boonthanome Nouanesengsy, Jonathan Woodring, John Patchett, Kary Myers, and James Ahrens. ADR visualization: A generalized framework for ranking large-scale scientific data using analysis-driven refinement. In *Large Data Analysis and Visualization (LDAV), 2014 IEEE 4th Symposium on*, pages 43–50. IEEE, 2014.

[37] Philippe Pébay, David Thompson, and Janine Bennett. Computing contingency statistics in parallel: Design trade-offs and limiting cases. In *Proceedings of the IEEE International Conference on Cluster Computing*, 2010.

[38] Hanspeter Pfister, William E. Lorensen, William J. Schroeder, Chandrajit L. Bajaj, and Gordon L. Kindlmann. The transfer function bake-off (panel session). In *IEEE Visualization*, pages 523–526, 2000.

[39] Bernice E. Rogowitz, Lloyd A. Treinish, and Steve Bryson. How not to lie with visualization. *Computers in Physics*, 10(3):268–273, 1996.

[40] D. Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends in Theoretical Computer Science*, 5(2):73–205, 2009.

[41] R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal of Computing*, 25:647–668, 1996.

[42] Ronitt Rubinfeld. Sublinear time algorithms. *International Conference of Mathematicians*, 2006.

[43] A. Salinger, K. Devine, G. Hennigan, H. Moffat, S. Hutchinson, and J. Shadid. MPSalsa: A finite element computer program for reacting flow problems, part 2 – users guide. Technical Report SAND96-2331, Sandia National Laboratories, September 1996.

[44] M. Salloum, J. Bennett, A. Pinar, A. Bhagatwala, and J. Chen. Enabling adaptive scientific workflows via trigger detection. 2015.

[45] C. Seshadhri, Ali Pinar, and Tamara G. Kolda. Triadic measures on graphs: The power of wedge sampling. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 10–18, 2013.

[46] C. Seshadhri, Ali Pinar, and Tamara G. Kolda. Wedge sampling for computing clustering coefficients and triangle counts on large graphs. *Statistical Analysis and Data Mining*, 7(4):294–307, 2014.

[47] R. Shan, C. S. Yoo, J. H. Chen, and T. Lu. Computational diagnostics for n-heptane flames with chemical explosive mode analysis. *Comb. Flame*, 159:3119–3127, 2012.

[48] Rick Stevens, Andrew White, Sudip Dosanjh, Al Geist, Brent Gorda, Kathy Yelick, John Morrison, Horst Simon, John Shalf, Jeff Nichols, and Mark Seager. Architectures and technology for extreme scale computing. Technical report, 2009.

[49] Jamaludin Suhaila and Abdul Aziz Jemain. Fitting daily rainfall amount in peninsular malaysia using several types of exponential distributions. *Journal of Applied Sciences Research*, 3(10):1027–1036, 2007.

[50] D. Thompson, J. Bennett, C. Seshadhri, and A. Pinar. A provably-robust sampling method for generating colormaps of large data. In *Large-Scale Data Analysis and Visualization (LDAV), 2013 IEEE Symposium on*, pages 77–84, Oct 2013.

[51] V. Vishwanath, M. Hereld, and M.E. Papka. Toward simulation-time data analysis and i/o acceleration on leadership-class systems. In *Proc. of IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, October 2011.

[52] P. M. Will and K. S. Pennington. Grid coding: a preprocessing technique for robot and machine vision. In *Proc. Int. Joint Conf. on Artificial Intelligence*, pages 66–70, 1971.

[53] C. S. Yoo, R. Sankaran, and J. H. Chen. Three-dimensional direct numerical simulation of a turbulent lifted hydrogen jet flame in heated coflow: Flame stabilization and structure. *Journal of Fluid Mechanics*, 640:453–481, 2009.

[54] Hongfeng Yu, Chaoli Wang, Ray W. Grout, Jacqueline H. Chen, and Kwan-Liu Ma. In-situ visualization for large-scale combustion simulations. *IEEE Computer Graphics and Applications*, 30:45–57, 2010.

DISTRIBUTION:

1	MS 0899	Technical Library, 8944 (electronic copy)
1	MS 0359	D. Chavez, LDRD Office, 1911

Sandia National Laboratories